Daniel Esteban Ramos Jiménez

Andrés Felipe Parra Quiroga

**Conociendo**

**Punto 1**

1. Tiene 3 paquetes
2. Tenemos 4 componentes
3. Son las clases concretas(4 en total), interface(solo 1), abstract(Solo 1) y los paquetes (2 en total).
4. El paquete de presentación tiene como objetivo ser el principal responsable de visualizar la aplicación de forma correcta, este ejecuta de manera adecuada la interfaz gráfica que le va a permitir al usuario interactuar
5. El paquete de dominio tiene como objetivo ser el que maneja las funcionalidades o comportamientos del aplicativo, como por ejemplo el comportamiento de las células

**Punto 2**

1. Para ejecutar cualquier método en java tenemos que ejecutar el main
2. El autómata lo encontramos en la clase de “automataGUI”

**Punto 3**

1. El programa en realidad no ofrece funcionalidades actualmente, lo único que se le puede hacer es visualizar las cuadriculas, además de tener la posibilidad hace clic en el botón de “TicTac” pero este botón aun no hace nada.
2. Actualmente no hace nada más que dejar visualizar las cuadriculas sin poder interactuar con ellas.
3. Esto es debido a que falta la implementación da algunas partes del código, las cuales son las encargadas de hacer que funciona correctamente el programa.

**Arquitectura General**

**Punto 1**

1. Un Paquete en Java es un contenedor de clases que permite agrupar las distintas partes de un programa y que por lo general tiene una funcionalidad y elementos comunes, definiendo la ubicación de dichas clases en un directorio de estructura jerárquica.
2. Los paquetes sirven para organizar los diferentes componentes de la aplicación, de tal forma que sea más fácil agruparlos por funcionalidades, encontrar que componente tiene x utilidad
3. En esta aplicación agrupamos dos paquetes principales, uno donde se ejecuta toda la GUI o la interfaz, y otro paquete el cual se encarga de las funcionalidades como tal del programa, como por ejemplo, el funcionamiento de las células, tanto de forma individual como en conjunto

**Punto 2**

1. Los paquetes y directorios son similares, las coincidencias en este caso es que cada paquete tiene un directorio asociado, de esta forma encontramos que hay dos paquetes, los cuales tiene un directorio asociado, dentro del directorio se presentan las clases contenidas dentro del paquete, junto a su documentación y a su ejecutor.

**Punto 3**

1. Revisar el .astah

**Arquitectura Detallada**

**Punto 1**

**Ciclo 1. Iniciado con las células normales**

1. A) Para almacenar los elementos se usa una lista estática, es decir, un array. B) Si puede recibir células. C) Porque mediante el método “setElemento” se puede agregar células, gracias a que la célula es un elemento.
2. A) Al momento de creas la célula esta se crea en estado “DESCONOCIDO”, debido a que usa parte del código de “Ser”, y en esta parte podemos observar como todo ser se crea en estado “DESCONOCIDO”. B) para pintarse usa una librería externa llamada “Color”, lo podemos notar cuando se importa la librería en la clase, además cuando se hace algún cambio de color o se define su color se llama a la librería también. C) La edad de la célula aumenta se ejecuta el método “cambie()”, porque podemos notar que es la única parte del código donde la edad se ve afectada.
3. La clase célula es definida gracias a la clase “Ser” y “Elemento”.
4. A) las células en general saben crearse en un autómata celular, entregar su estado actual, en cuando a la fila, columna en la que esta posicionada y entregar que color es actualmente, puede cambiar su estado actual a su estado siguiente, y para esto tiene que decidir cual va a ser su siguiente estado, de igual forma tiene capacidad para saber si esta viva o muerta, aumentar su edad, obtener su edad y saber si esta viva o muerta en su estado actual. B) Cosas como la edad que cumple o saber cuando tiene que cambiar de estado son cosas que se van a mantener constantes durante el desarrollo del ejercicio. C) Si bien la célula sabe cuándo debería cambiar de estado este cambio no se efectúa correctamente puesto que le hace falta el código actualmente, además le falta la parte en que si una célula cambia de estado las células a su alrededor también tendrán que cambiar de estado.

datos: Estado (char), edad (int)

que ya sabe hacer: cumple(), getEdad(), isVivo()

que decide hacer distinto:Noneque no puede hacer distinto:cumple()

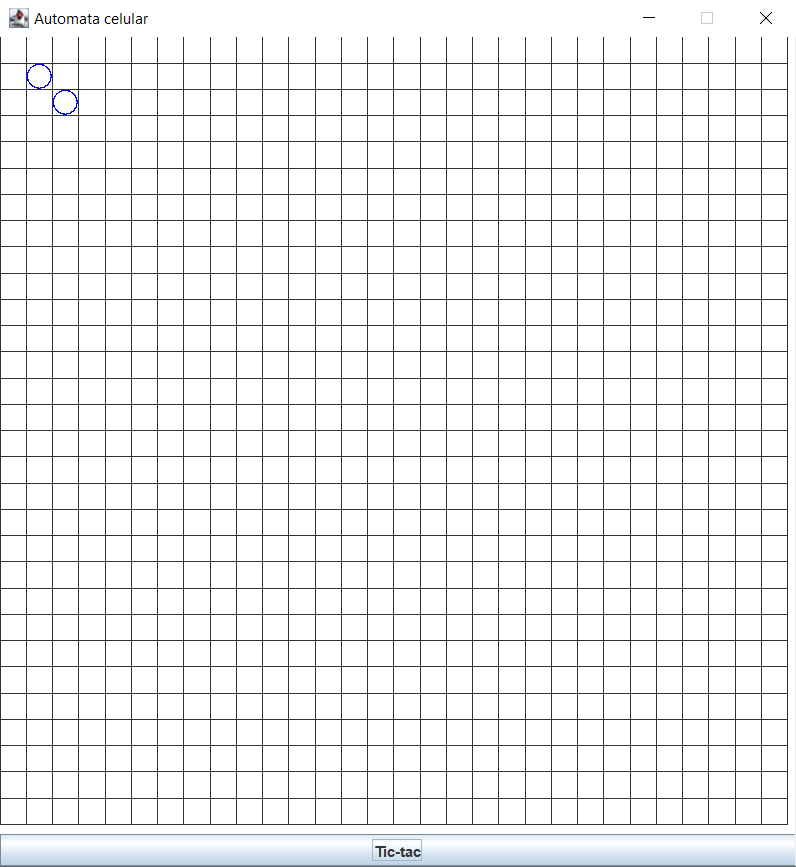
1. datos: Forma

que ya sabe hacer: forma(), getColor()

que decide hacer distinto: decida(), getColor()

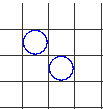
que no puede hacer distinto: None

1. Una vez implementado el código podemos observar como quedan las dos células en la posición (1,1) y (2,2).

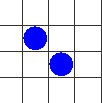
****

1. Ambas células se crean en el mismo instante de tiempo, por lo que el avance de ambas alrededor del tiempo va a ser el mismo. En este caso al primer TicTac ambas células van a tener edad 1, en el segundo TicTac ambas células van a tener edad 2, finalmente al tercer TicTac las células van a tener edad 3. Desde el estado 1, hasta el estado 3 del TicTac ambas células se mantienen vivas en todo momento.
2. El método funciona correctamente como se esperaba, descrito en el punto 8.

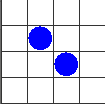
TicTac 0:

****

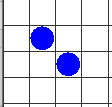
TicTac 1:

****

TicTac 2:

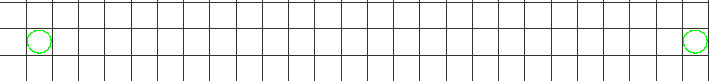
****

TicTac 3:

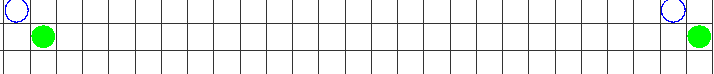
****

**Ciclo 2. Incluyendo a las células especiales**

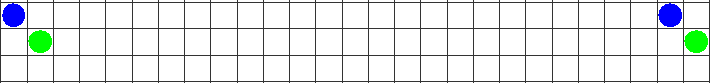
1. Solo se sobrescribe un método, el de decida.
2. Como agregamos a agamenon en una parte central y a venus en una esquina, agamenon quedaría al pimer tic con una célula normal en estado desconocido, al igual que venus. En el segundo tic agamenon tiene solo la anterior célula si no que esta vez ya esta en estado viva y con edad 1, al igual que venus. Al tercer click agamenon quedaría con una célula viva de edad 2, al igual que venus.
3. El método construido funciona de manera correcta.
4. Tic 0:

****

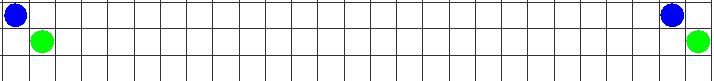
Tic 1:



Tic 2:



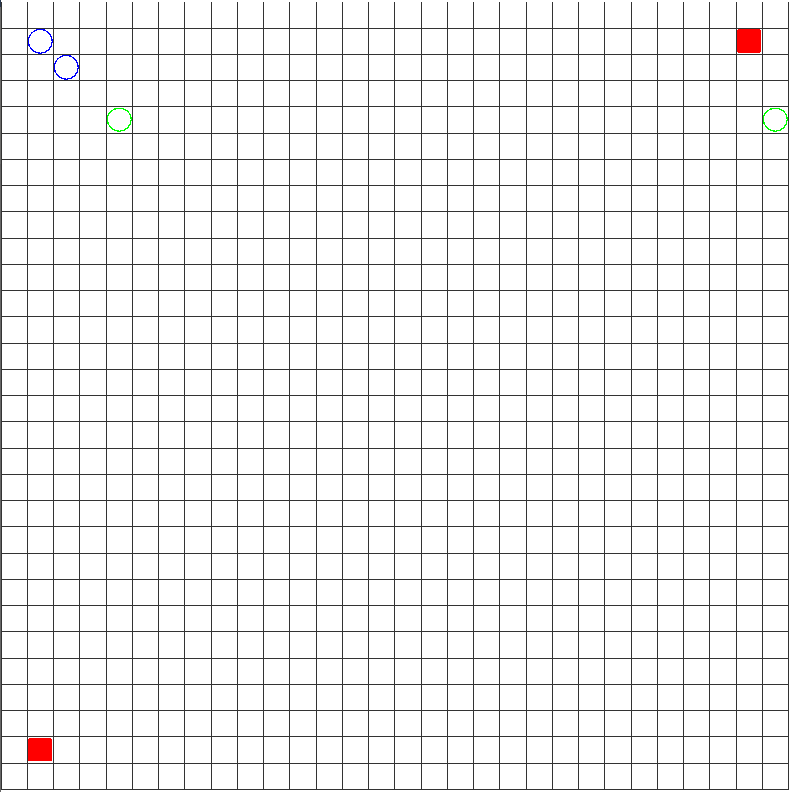
Tic 3:



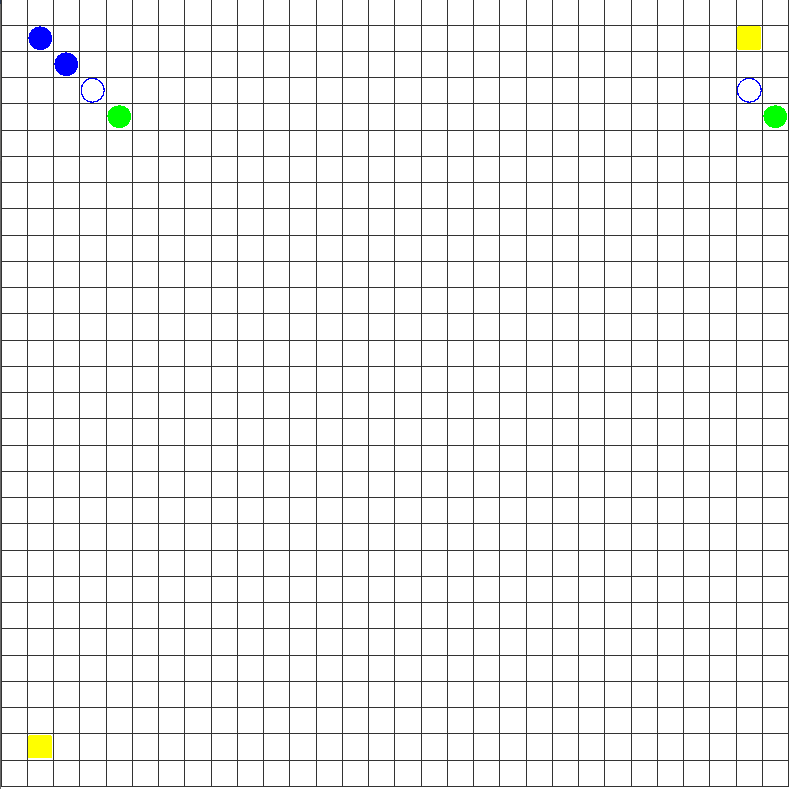
**Ciclo 3. Adicionando una mini-calefactores**

1. Para crear un calefactor lo primero que hacemos es crear esta clase, lo principal que se puede ver es que esta clase implementa la interface Elemento, de esta forma nos aseguramos de que no sea necesario cambiar el código ya realizado de AutomataCelular, puesto que el TicTac no solo trabaja con células, si no que trabaja con Elementos.
2. La pregunta está incompleta.
3. Los calefactores al tic 1 serian amarillos, al tic 2 rojos y finalmente en el tic 3 serían amarillos nuevamente.
4. Funciona de manera correcta el método.

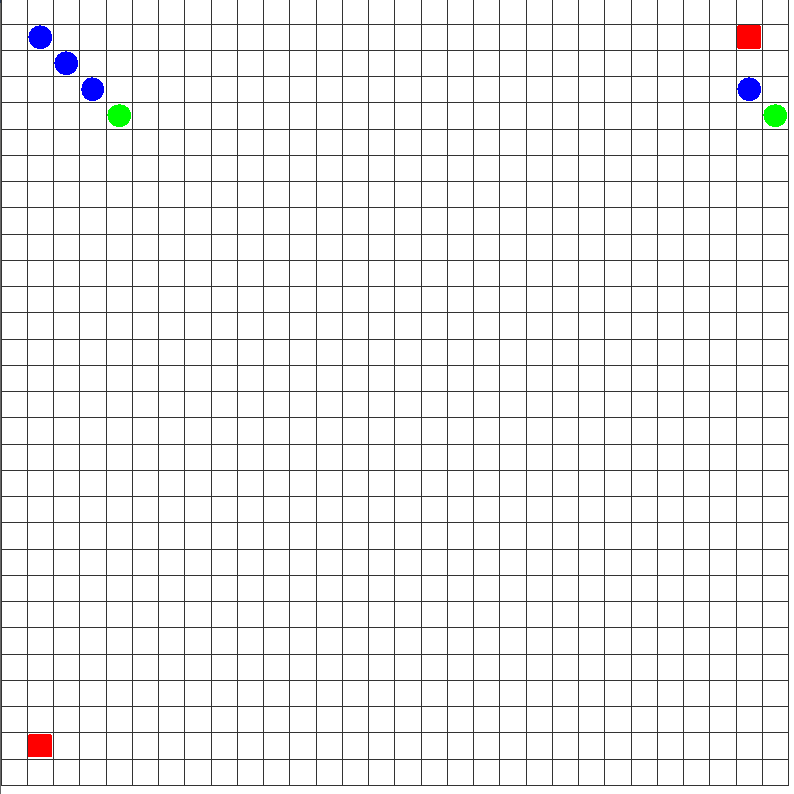
Tic 0:

****

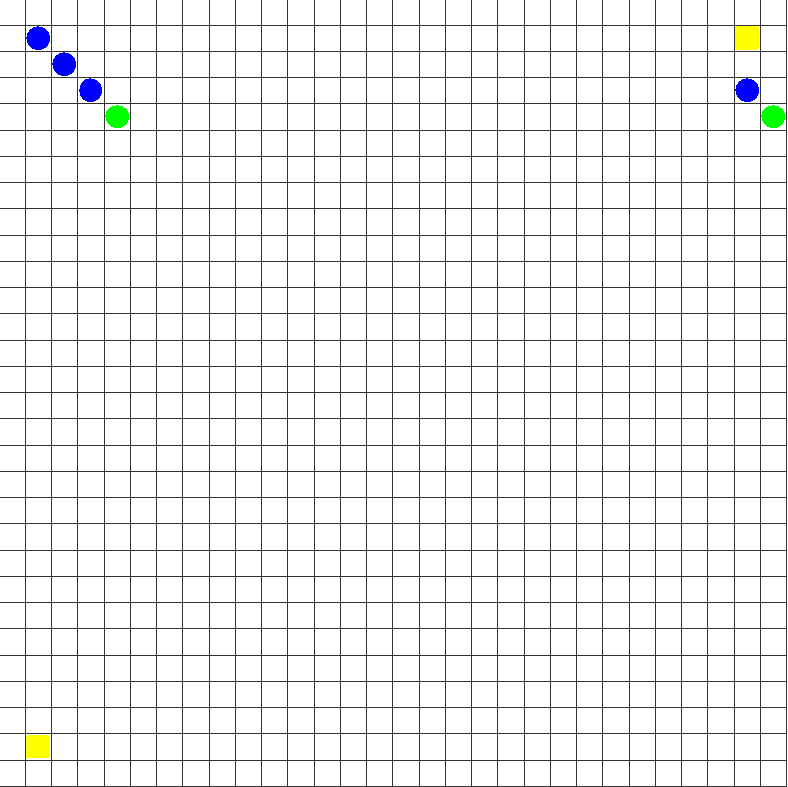
Tic 1:



Tic 2:



Tic 3:



**Ciclo 4. Nueva Célula**

La nueva célula va a ser una simulación de una cepa zombie las características principales de estas van a ser:

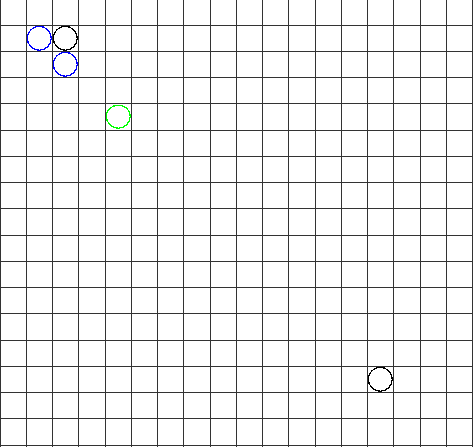
* La nueva célula convertirá a todas las células (vivas o muertas) en una de su tipo
* Estas células morirán después de tres ejecuciones si no infectan a otras células (cada vez que una célula zombie infecta a una célula normal su tiempo de vida se reestablece)
* Las nuevas células zombies se generan en estado “DESCONOCIDO”

Para el primer caso significativo vamos a tomar la célula zombie andresParra, la cual estará ubicada en la posición (1,2), al lado de él se podrá encontrar indiana y 007, cerca de el también hallamos a agamenon.

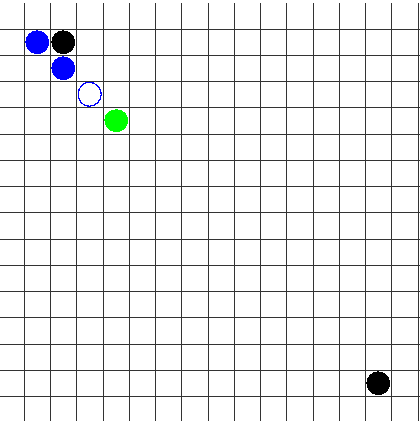
Para el segundo caso significativo encontraremos a danielRamos en la posición (14,14), ni cerca ni al lado de el hay ninguna otra célula.

Los casos significativos que escogimos representan como es que funciona correctamente la infección zombie, como si aislamos a una de las células va a morir después de 3 Tics, en cambio la que esta rodeada será capaz de infectar a las que estén a su alrededor. Podemos notar estos casos en las siguientes imágenes.

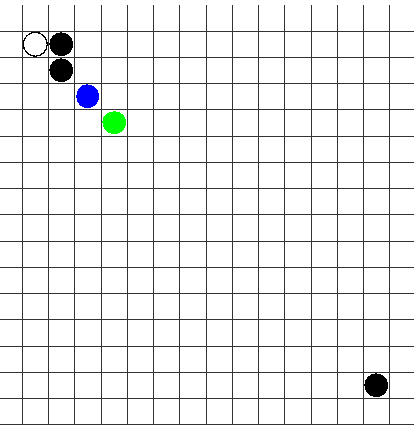
Tic 0:



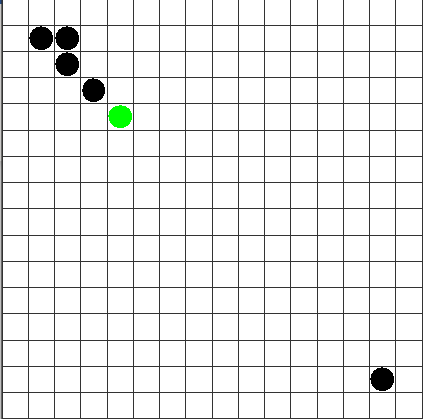
Tic 1:



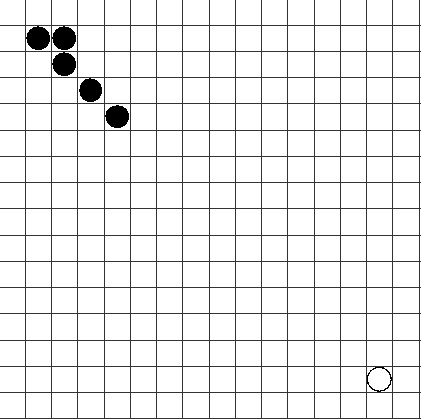
Tic 2:



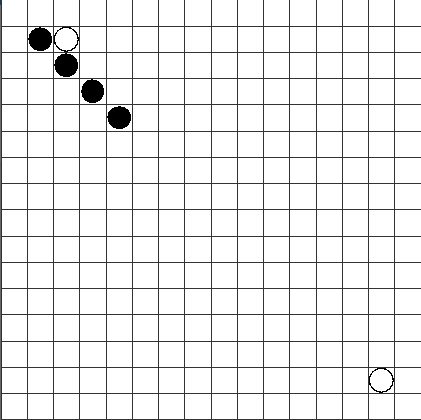
Tic 3:



Tic 4:



Tic 5:



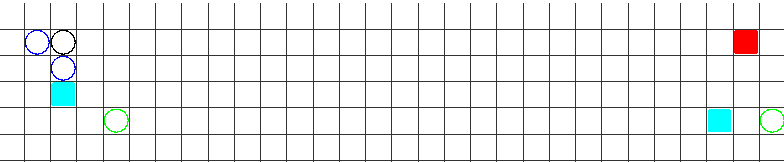
**Ciclo 5. Nuevo Elemento**

El nuevo elemento estará relacionado con la célula previamente creada, este nuevo elemento será una fortaleza, las fortalezas tendrán como objetivo matar las células zombies que se encuentren cerca en un radio de 2 cuadrados. Cualquier otra célula que no sea zombie no se vera afectada por el ataque de la fortaleza. Las fortalezas serán de color cyan y con forma cuadrada.

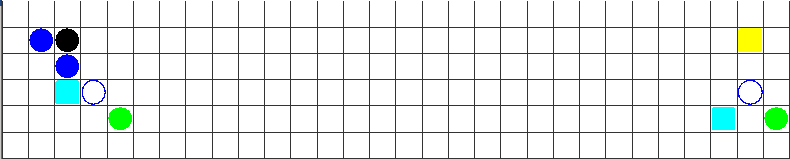
El primer caso significativo lo tenemos marineFord, una fortaleza la cual esta ubicada cerca de la cepa zombie de andresParra, en este caso podremos ver como la fortaleza evita que la cepa llegue hasta agamenon, dejando esta ultima viva algunos tics mas, hasta que su periodo de vida se complete.

El segundo caso significativo lo encontraremos en la fortaleza eniesLobby, la cual se encuentra a unas cuantas casillas de venus, lo que se busca resaltar en este caso es como la fortaleza solo ataca a las células zombies, de tal forma que si una célula no es zombie no será atacada.

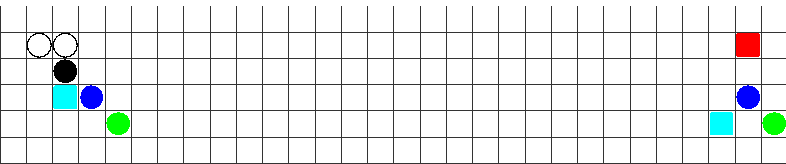
Tic 0:



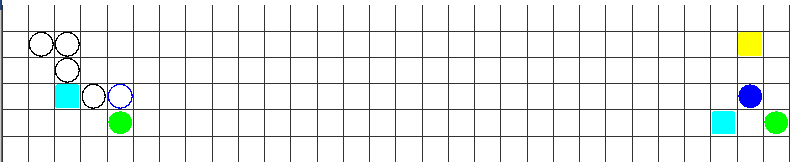
Tic 1:



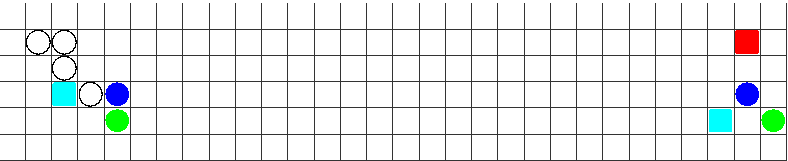
Tic 2:



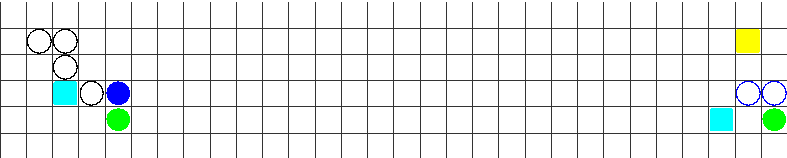
Tic 3:



Tic 4:



Tic 5:



**Ciclo 6. El juego de la vida**

**Retrospectiva**

1. 2:30/Daniel Ramos, 2:30/Andres Parra
2. El laboratorio tiene realizada hasta la parte 2, nos falta el ultimo punto, teníamos bastantes problemas con algunos puntos del laboratorio
3. Es importante para completar en totalidad las entregas en el tiempo destinado, de esta forma evitando cuellos de botella, además no se depende del trabajo de las demás personas
4. Conseguir Visualizar ll cuadricula dentro de BlueJ, porque tuvimos bastantes problemas para visualizarlo al inicio, además fue un logro conseguir sacar rápido del diagrama de paquetes
5. El trabajo en equipo fue bueno, nos falta mejorar un poco los tiempos de trabajo y reducir las distracciones.