JUEGO DE LA VIDA

PRÁCTICA 2

En esta tarea nos han pedido realizar “El juego de la vida”, fue publicado en 1970, por el matemático Horton Conway.

Éste juego es un buen ejemplo de emergencia y autoorganización.

PABLO Y EDURNE

FUNCIONAMIENTO:

Para comenzar el desarrollo del programa, es necesario un tablero de unas dimensiones introducidas por el usuario (filas, columnas), controlado por un método **"comprobarEntrada"** que evita que los valores introducidos puedan generar error. Con ello se crea un **Array bidimensional de String**; que debe ser rellenado aleatoriamente de un carácter ("*" -> célula, o " " -> no célula). Para ello se ha creado un método **"rellenar"**, que recorre las filas y columnas de la tabla, y escribe en la celda, con un 20% de probabilidad una célula, o en lo contrario quede vacía, con la función **"Math.random".**

También se pide al usuario que introduzca un número de generaciones, que van a ser las veces que se desarrolle el proceso principal.

A su vez, se ha creado un ArrayList de la clase **"Tripleta",** que contiene los siguientes atributos:

- **Generación** ---> Corresponde a la iteración en la que se encuentra el proceso.
- **Células vivas** ---> Corresponde al número de células vivas que hay en esa generación.
- **Células nuevas** ---> Corresponde al número de células nuevas que se han creado en esa generación. (Es la resta de las células vivas actuales, menos las vivas de la generación anterior).

Se crean dos constructores del objeto; uno general y otro que recibe los parámetros de células vivas y células nuevas; la generación se va creando automáticamente cada vez que se genera un objeto nuevo. A su vez, se crean los métodos **"Getter y Setter"** correspondientes.

Para los conteos de células, se crea un método en la clase principal, que consiste en hacer un sumatorio de células que están presentes.

El desarrollo que se repite en cada iteración es el siguiente:

1. Se establece un conteo del número de células actuales y se añaden al **ArrayList.**
2. Se invoca al método "**visualizar**" que nos muestra el tablero por pantalla.
3. Se invoca al método "**supervivencia"**. Que realiza el proceso principal del juego.
   Se crea una variable boolean de "**estado**", que controla si la célula que se comprueba está viva o muerta.
   Con dos bucles for anidados se hace un conteo sobre las 8 celdas de alrededor, implementando también un método "**validacionCelda**", que evita que se salga de los límites del tablero y genere un error.
   Finalmente, con el dato del estado y del conteo de células de alrededor, se establecen unos condicionales que contemplan todos los casos de supervivencia, muerte o resucitación que puede suceder en la casilla. Se aplica a una **tabla auxiliar** para no variar la original y generar error en los conteos.
   Cuando ha finalizado de recorrer la tabla original, se vuelca la tabla auxiliar a la original y finaliza el método, habiendo realizado una nueva generación.
4. Hay un condicional para comprobar si todavía hay células vivas; en caso contrario, saldrá del bucle.

5. Volvería a comenzar el bucle en el punto 1, añadiendo la información de la generación que se ha creado en el punto 3 anteriormente.

Por último, el programa muestra la información que se nos pide con un resumen de lo acontecido en el desarrollo.

PROGRAMA DE PRUEBAS:

Se nos pide desarrollar una serie de tableros predefinidos, que no requieren interacción con el usuario. Se ha dividido el programa de prueba en tres clases diferentes para cada tablero.

Para cada clase, se genera un tablero con un patrón, estableciendo las 30 generaciones y el número de celdas, aplicando los métodos de contar células, visualizar tablero y supervivencia de las células, que están contenidos en la clase **Main** del programa principal.

- El primer supuesto, llamado "**bloque**", es un cuadrado que no varía en ninguna de sus iteraciones.

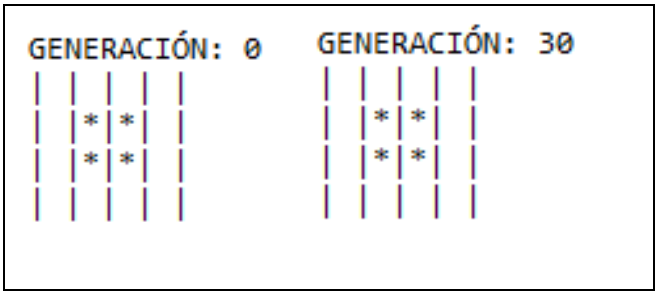

Figura 1. Generaciones inicial y final del patrón "bloque".

- En el segundo supuesto, llamado "**intermitente**", van variando las células de los laterales, generando la impresión de un molino que gira.

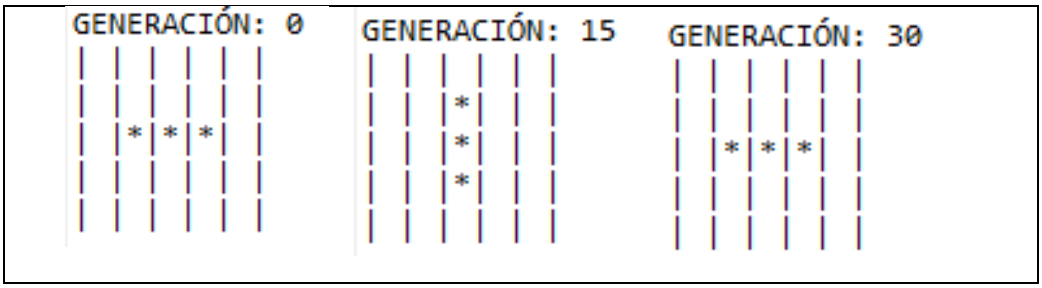

Figura 2. Generaciones iniciales, intermedia y final, del patrón "intermitente".

- En el tercer supuesto, llamado “**faro**”, dos posiciones se van creando y destruyendo de forma infinita:

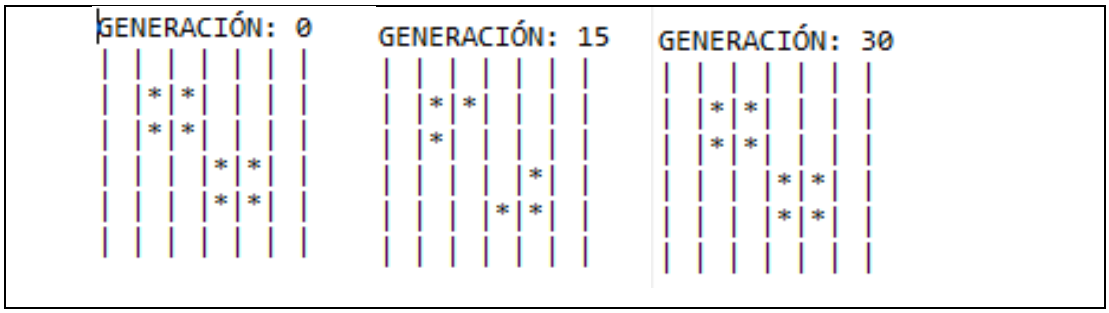

Figura 3. Generaciones iniciales, intermedia y final, del patrón “faro”: