Proyecto 2 UT5 (para hacer en casa y entregar en GitHub)

Objetivos

- *definir arrays bidimensionales*
- *utilizar arrays de dos dimensiones*
- *utilizar bucles for simples y anidados*
- *procesar submatrices*
- *utilizar arrays de tres dimensiones*
- *utilizar la clase Arrays (equals, copyOf, fill, toString)*

Antes de empezar

- Este ejercicio es para realizar de forma individual en casa.
- El proyecto de partida está en https://github.com/aetxabao/GameOfLife . Deberás hacer un *fork* a tu cuenta y clonarlo en tu PC desde línea de comandos
- Una vez completado desde un terminal haz un *push* del último *commit* a GitHub
- Haz un Pull Request para que sea corregido antes del plazo límite
- Se valorará en la corrección que el programa esté probado (compila y ejecuta bien) y que esté claramente escrito y organizado (se respetan las reglas de estilo del lenguaje Java, nombres descriptivos, comentarios adecuados, código no duplicado, ...)

- La fecha tope de entrega es el **Domingo 16 de Enero** hasta las **23,30h**.

- Se anulará automáticamente la corrección del ejercicio y se evaluará con un 0 si se detecta que ha sido copiado o dejado copiar a algún compañero/a
- Se penalizará si no se siguen las normas de entrega del ejercicio
  - ✗ no se ha hecho un fork / no se ha hecho un *pull request*
  - ✗ hay algún *commit* con posterioridad a la fecha de entrega
  - ✗ el *pull request* se hace después de la fecha de entrega
- El profesorado podrá convocar al alumno/a para defender oralmente el proyecto

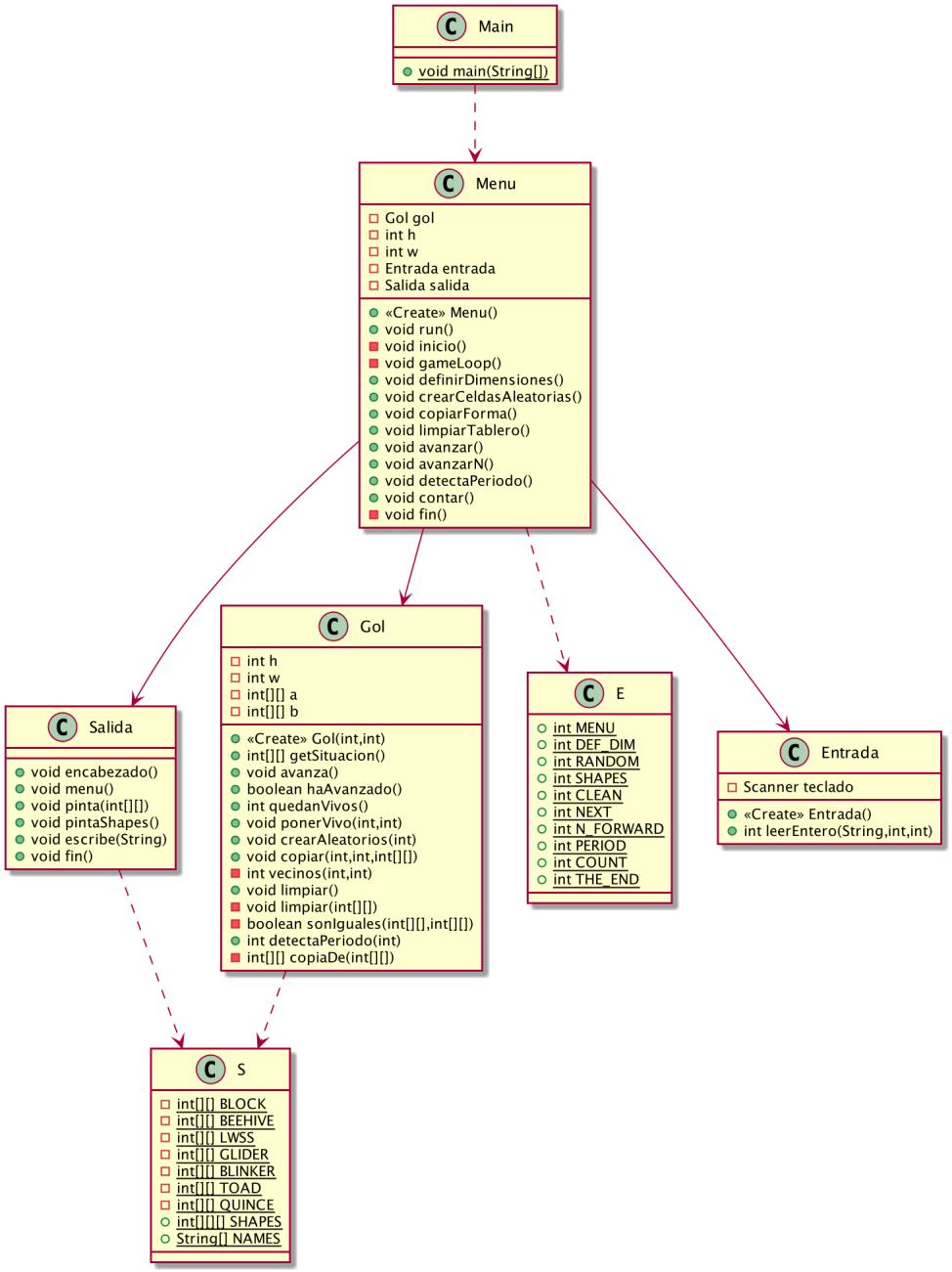
Especificaciones

En este proyecto principalmente vamos a completar la clase Gol que es la base para ejecutar lo que es conocido como el "Juego de la vida" o "Game Of Life" en inglés.

Haz el *fork* del proyecto **GameOfLife** desde https://github.com/aetxabao a tu cuenta GitHub y desde un terminal clona el proyecto a tu PC.

Abre el proyecto en IntellijIDEA. Tienes que completar las clases `Gol` y `Salida`.

No olvides escribir tu nombre después de la etiqueta *@author* en todas ellas



La clase Main hace uso de una instancia de Menu que contiene tres referencias a instancias de Salida, Gol y Entrada. La clase Gol tiene como atributos las dimensiones del tablero ancho(w) x alto(h) y dos referencias a dos matrices que se utilizan para el desarrollo del juego. La clase Menu hace uso de las constantes de la clase E para gestionar las opciones del menu y las clases Salida y Gol utilizan la clase S que contiene un conjunto de matrices.

El juego [1]

Se trata de un juego de cero jugadores, lo que quiere decir que su evolución está determinada por el estado inicial y no necesita ninguna entrada de datos posterior. El "tablero de juego" es una malla plana formada por cuadrados (las "células") que se extiende por el infinito en todas las direcciones. Por tanto, cada célula tiene 8 células "vecinas", que son las que están próximas a ella, incluidas las diagonales. Las células tienen dos estados: están "vivas" o "muertas" (o "encendidas" y "apagadas" o valores “1” y “0”). El estado de las células evoluciona a lo largo de unidades de tiempo discretas (se podría decir que por turnos). El estado de todas las células se tiene en cuenta para calcular el estado de las mismas al turno siguiente. Todas las células se actualizan simultáneamente en cada turno, siguiendo estas reglas:

•Una célula muerta con exactamente 3 células vecinas vivas "nace" (es decir, al turno siguiente estará viva).

•Una célula viva con 2 o 3 células vecinas vivas sigue viva, en otro caso muere (por "soledad" o "superpoblación").

clase `Gol`

El tablero tiene una matriz de enteros cuyos valores representan si una celda está tapada todavía o si se ha descubierto el número de bombas que tienen en las celdas colindantes o si es una bomba.

Completa los métodos siguientes:

- método **constructor –** inicializa los atributos teniendo en cuenta que h es la altura (número de filas) y w es el ancho (número de columnas).
- método **getSituacion** - devuelve la matriz principal a que representa la situación actual.
- método **ponerVivo –** asigna el valor 1 a la celda de la matriz principal a fila f y columna c.
- método **crearAleatorios** - pone n posiciones aleatorias con valor 1.
- método **vecinos –** calcula el número de celdas vecinas vivas (no se considera así misma).
- método **quedanVivos** - cuenta todas las celdas vivas del tablero.
- método **limpiar –** establece todos los valores de la matriz pasada como parámetro a 0.
- método **copiar –** copia en la matriz principal "a" a partir de la posición dada por la fila f y la columna c, los valores de la matriz "m" pasada como parámetro.
- método **copiaDe** - devuelve una nueva instancia con las dimensiones y los valores de la matriz pasada como parámetro.
- método **sonIguales –** comprueba si dos matrices tienen las mismas dimensiones y valores.
- método **avanza** - recorre todas las celdas de la matriz principal a y calcula el número de vecinos que tiene. En base a las reglas del juego de la vida en la matriz secundaria b se establecerá el valor que tendrá la celda en el siguiente instante de tiempo. Tras haber recorrido todas las celdas la matriz principal debe pasar a ser la secundaria y al revés, la secundaria la principal. Para ello se debe utilizar una referencia local a una matriz sin instanciar, por ejemplo "x", para que cuando "a" apunte a "b" la referencia de "a" no se pierda y apuntando "b" a "x" se hace el cambio. Es el mismo mecanismo para intercambiar el valor de dos variables primitivas utilizando una tercera variable local. Para que en la matriz secundaria b se vayan haciendo los cálculos es preciso limpiar la matriz b antes de empezar a recorrer las celdas.
- método **detectaPeriodo –** crea una copia local de la matriz principal "a" y avanza el estado hasta un número límite de veces comprobando cada vez si la copia local es igual a la matriz principal "a". En caso de ser iguales el método devuelve el numero de pasos transcurridos. Si se finaliza el bucle sin encontrar el periodo se devolverá Integer.MAX_VALUE y la matriz principal recuperará los valores de antes de empezar.

clases `Salida` y `S`

La clase S tiene matrices con algunas de las formas habituales del Juego de la vida y sus nombres [2]. El array SHAPES contiene a las matrices cuyos nombres están en el array NAMES.

Completa:

- método **pintaShapes –** escribe los nombres de las formas definidas en S y debajo los arrays que componen cada una de las matrices. Para la representación se establece un espacio de 15 caracteres por 5 filas para cada forma incluyendo su nombre. Todos deben aparecer seguidos como en la siguiente imagen y en el ejemplo al final.

```
0. BLOCK        1. BEEHIVE      2. LWSS         3. GLIDER       4. BLINKER      5. TOAD         6. QUINCE
[1, 1]          [0, 1, 1, 0]    [1, 0, 0, 1, 0] [0, 1, 0]       [1, 1, 1]       [0, 1, 1, 1]    [1, 1, 1, 1, 1, 1, 1, 1]
[1, 1]          [1, 0, 0, 1]    [0, 0, 0, 0, 1] [0, 0, 1]                       [1, 1, 1, 0]    [1, 0, 1, 1, 1, 1, 0, 1]
                [0, 1, 1, 0]    [1, 0, 0, 0, 1] [1, 1, 1]                                       [1, 1, 1, 1, 1, 1, 1, 1]
                                [0, 1, 1, 1, 1]
```

Rúbrica evaluación

clase `Gol`	
constructor	2
getSituacion	1
ponerVivo	1
crearAleatorios	5
vecinos	6
quedanVivos	5
limpiar	5
copiar	5
copiarDe	5
sonIguales	5
subtotal	40

clase `Gol`	
avanza	20
detectaPeriodo	20
subtotal	40

clase `Salida`	
pintaShapes	20
subtotal	20

Penalización

Falta de claridad del código 0,5 / 10.

Referencias

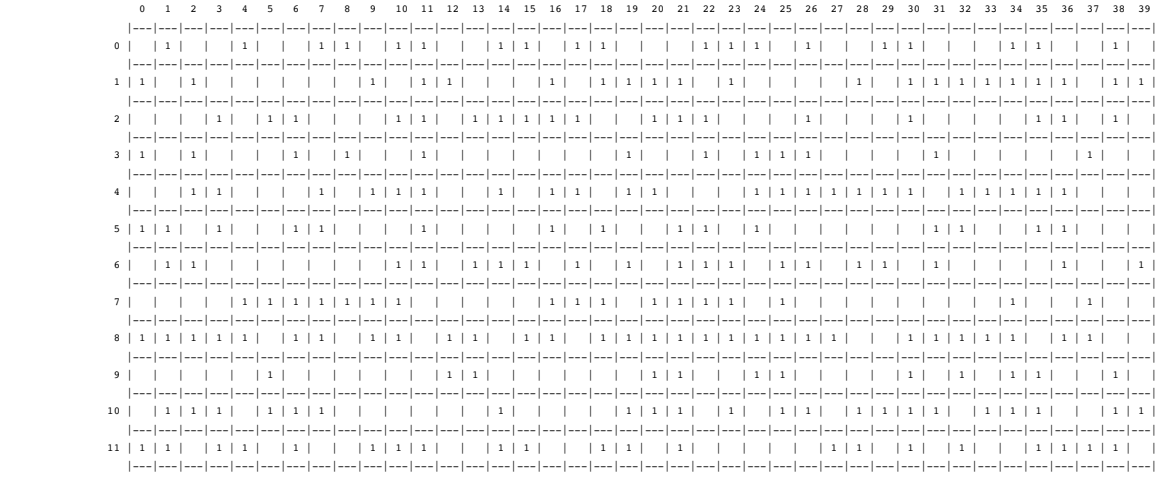
[1] Juego de la vida, Wikipedia.
https://es.wikipedia.org/wiki/Juego_de_la_vida

[2] El juego de la vida, Manuel Romero Dopico, UC3M.
https://www.it.uc3m.es/jvillena/irc/practicas/09-10/04mem.pdf

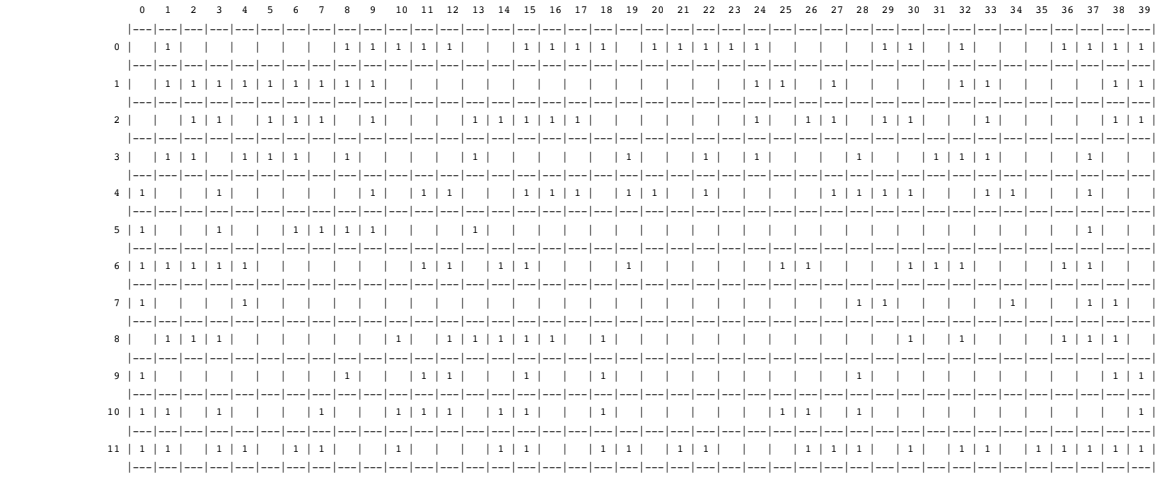
[3] Play John Conway's Game Of Life
https://playgameoflife.com/

[4] Ejemplo GameOfLife, aetxabao, YouTube.
https://youtu.be/-EJuD3B5YgI

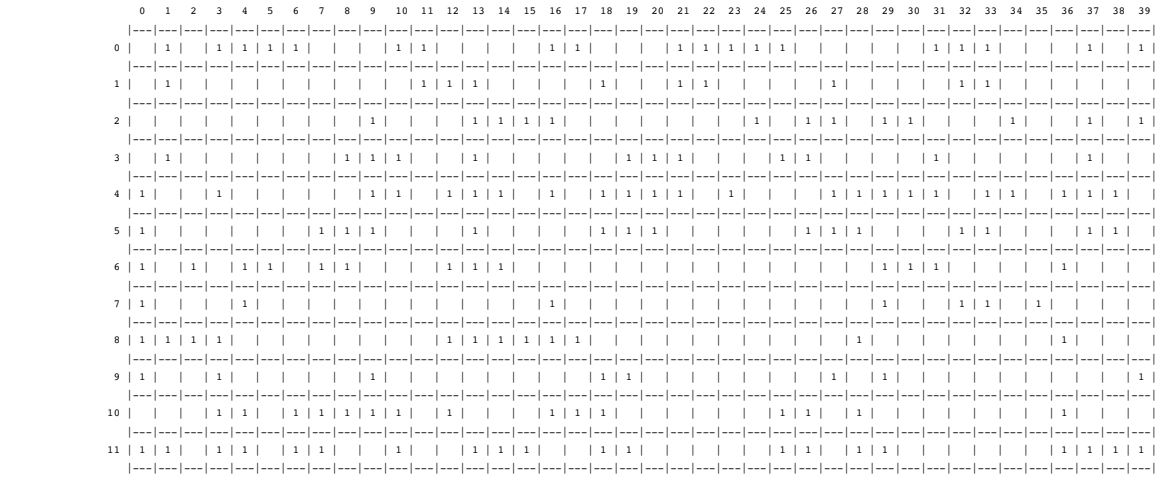
```
================================================
=          G a m e    O f    L i f e          =
================================================
1. Imprimir el menú
2. Definir dimensiones tablero
3. Crear celdas aleatorias
4. Copiar formas
5. Limpiar tablero
6. Avanzar un estado
7. Avanzar N estados
8. Buscar periodo
9. Contar seres vivos
0. Terminar
Opción [0-9]: 3
Numero de celdas [1-480]: 300
```



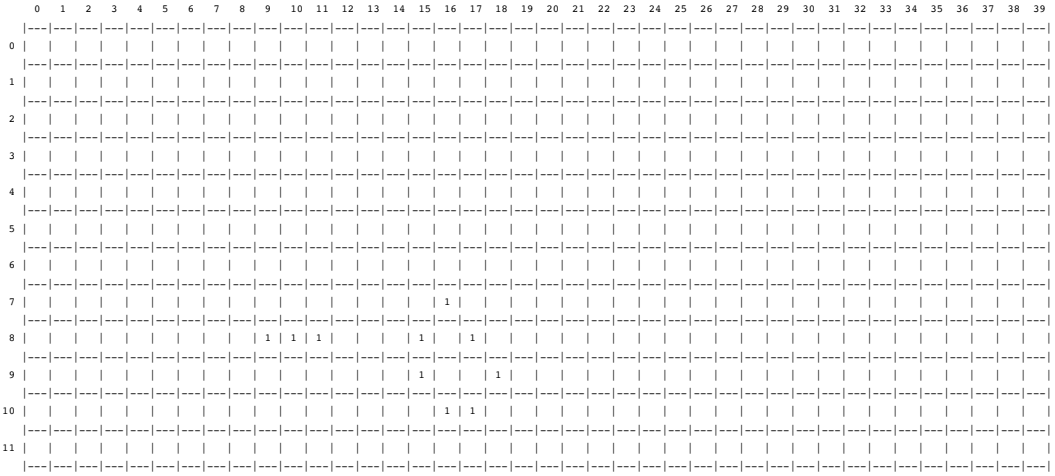
```
Opción [0-9]: 6
```



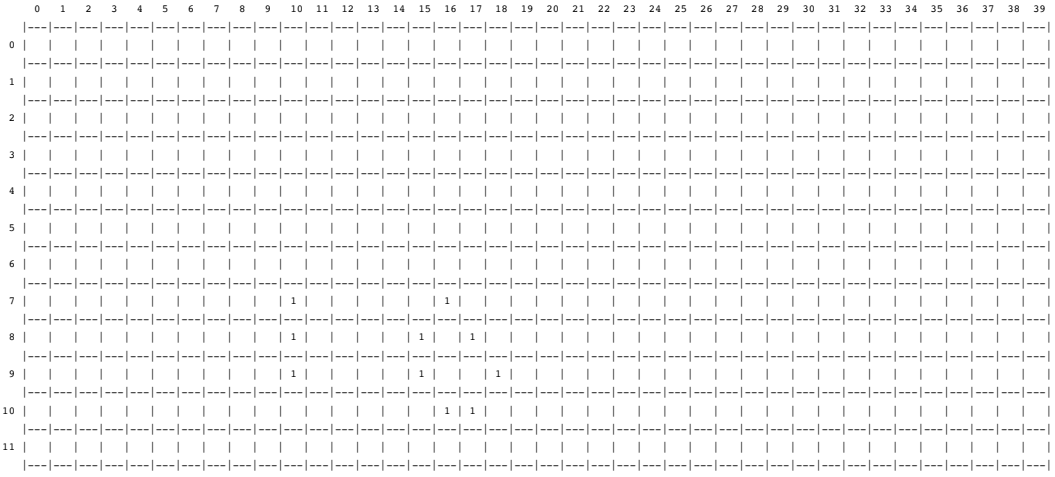
```
Opción [0-9]: 6
```



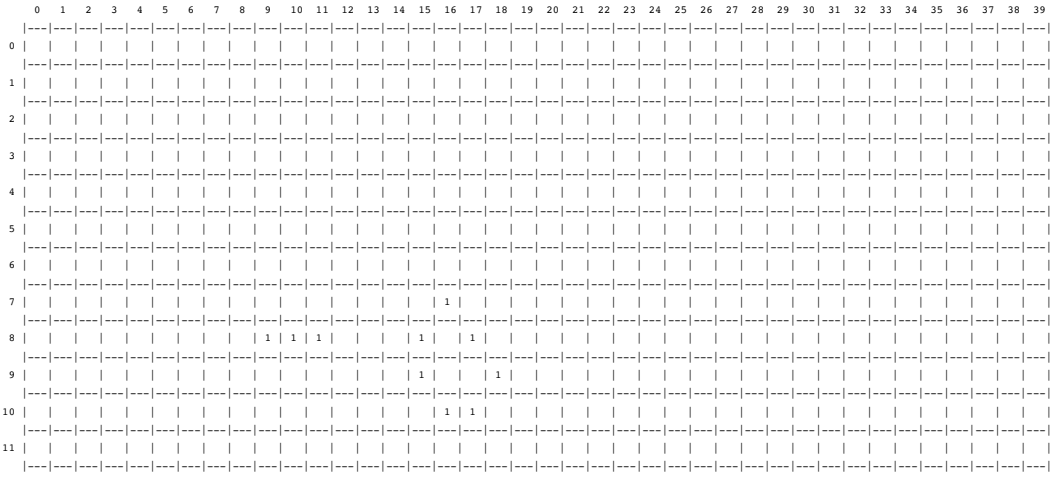
```
Opción [0-9]: 7
Cuanto [1-10000]: 1000
```



```
Opción [0-9]: 6
```

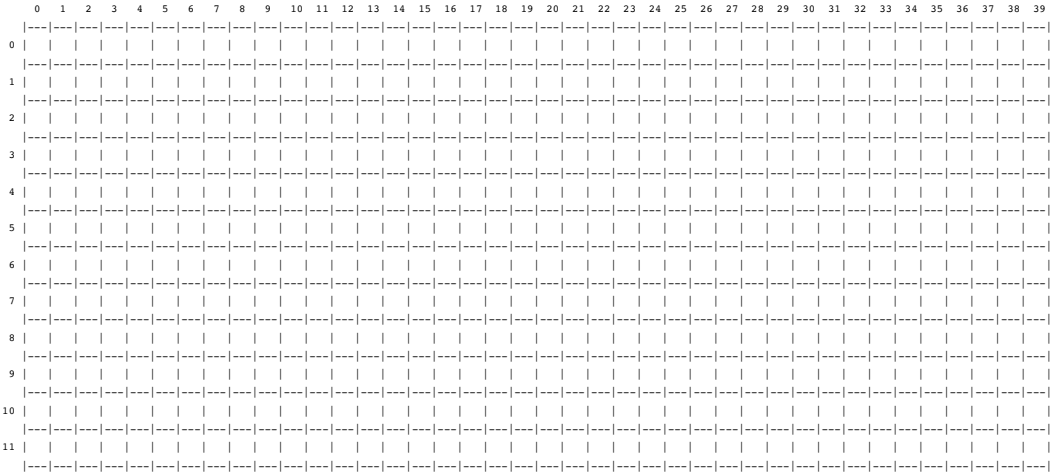


```
Opción [0-9]: 6
```

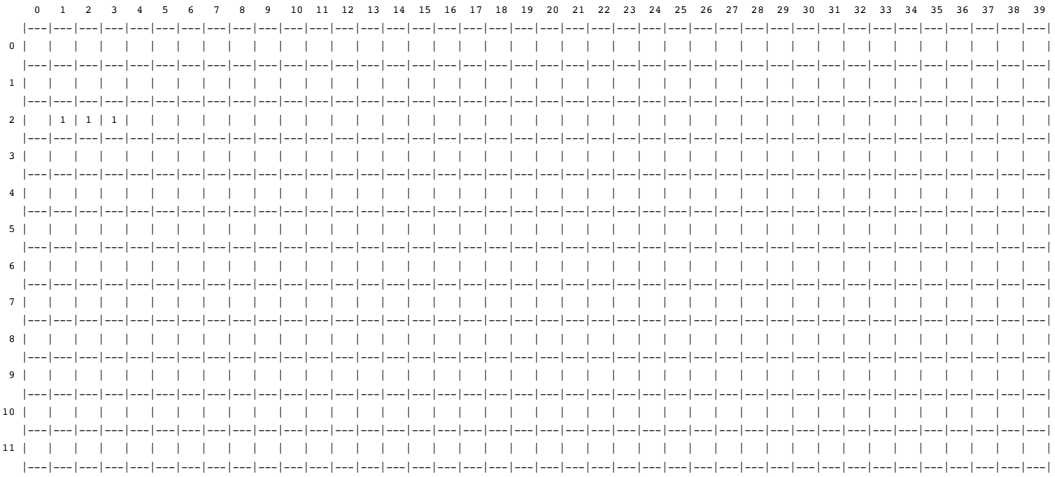


```
Opción [0-9]: 9
Quedan 10 vivos.
Opción [0-9]: 1
1. Imprimir el menú
2. Definir dimensiones tablero
3. Crear celdas aleatorias
4. Copiar formas
5. Limpiar tablero
6. Avanzar un estado
7. Avanzar N estados
8. Buscar periodo
9. Contar seres vivos
0. Terminar
Opción [0-9]: 5
```

```
Opción [0-9]: 4
0. BLOCK        1. BEEHIVE      2. LWSS         3. GLIDER       4. BLINKER      5. TOAD         6. QUINCE
[1, 1]          [0, 1, 1, 0]    [1, 0, 0, 1, 0] [0, 1, 0]       [1, 1, 1]       [0, 1, 1, 1]    [1, 1, 1, 1, 1, 1, 1, 1]
[1, 1]          [1, 0, 0, 1]    [0, 0, 0, 0, 1] [0, 0, 1]                       [1, 1, 1, 0]    [1, 0, 1, 1, 1, 1, 0, 1]
                [0, 1, 1, 0]    [1, 0, 0, 0, 1] [1, 1, 1]                                       [1, 1, 1, 1, 1, 1, 1, 1]
                                [0, 1, 1, 1, 1]
Figura [0-6]: 4
```

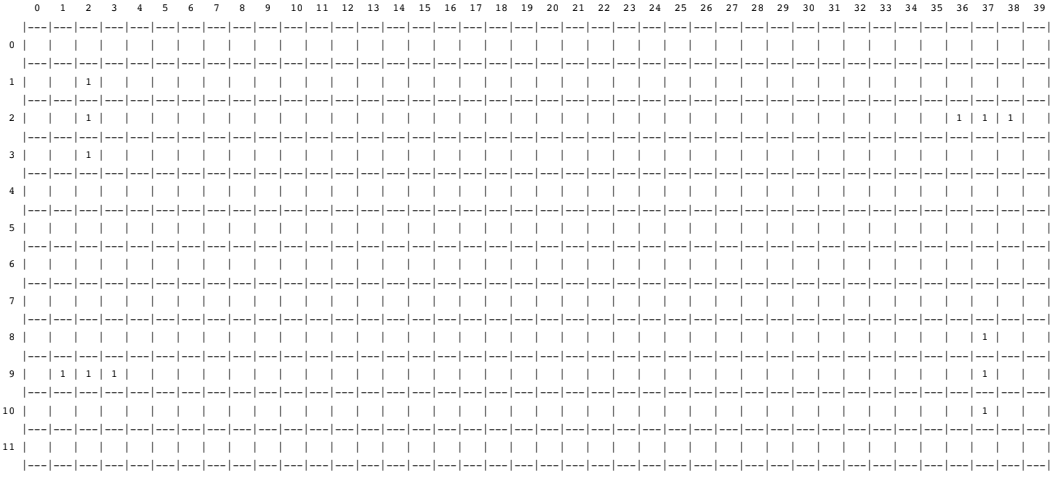


```
Numero de fila [0-11]: 2
Numero de columna [0-39]: 1
```



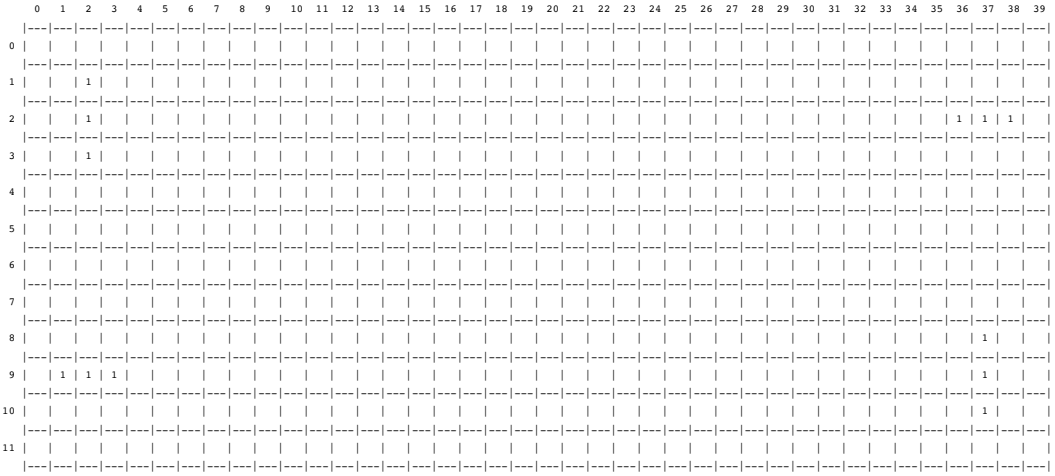
```
Opción [0-9]:

Tras introducir la siguiente serie de valores:
4, 4, 9, 36,     6,     4, 4, 9, 1,    4, 4, 2, 36
```

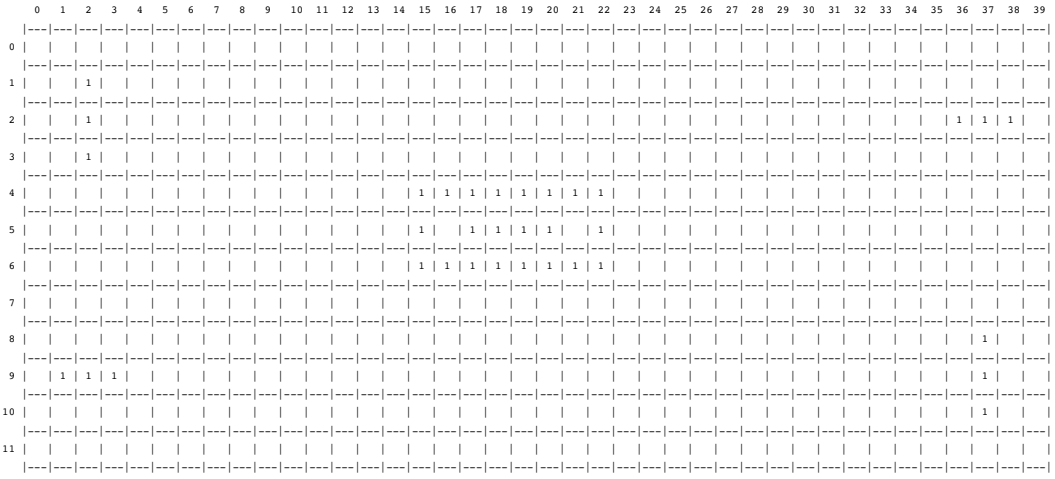


```
Opción [0-9]: 8
El periodo es 2
```

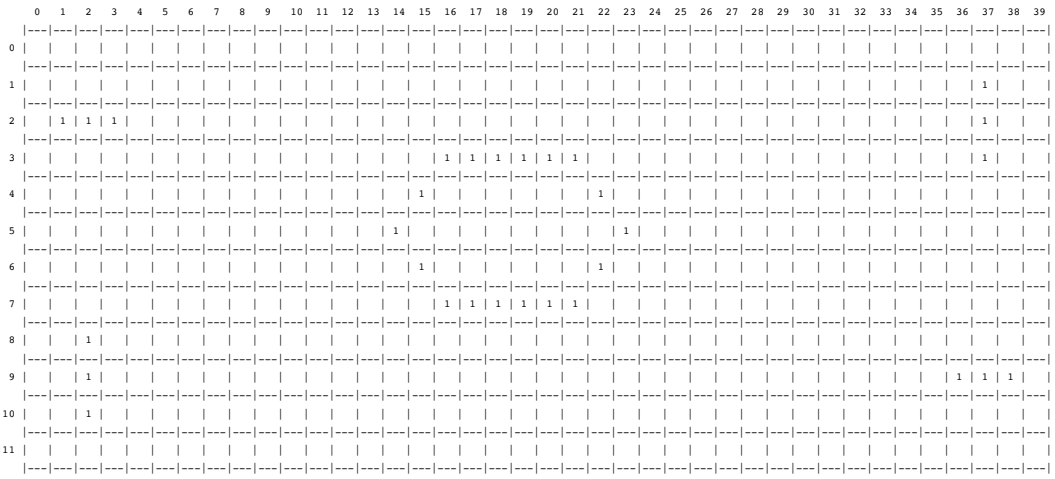
```
Opción [0-9]: 4
0. BLOCK        1. BEEHIVE      2. LWSS         3. GLIDER       4. BLINKER      5. TOAD         6. QUINCE
[1, 1]          [0, 1, 1, 0]    [1, 0, 0, 1, 0] [0, 1, 0]       [1, 1, 1]       [0, 1, 1, 1]    [1, 1, 1, 1, 1, 1, 1, 1]
[1, 1]          [1, 0, 0, 1]    [0, 0, 0, 0, 1] [0, 0, 1]                       [1, 1, 1, 0]    [1, 0, 1, 1, 1, 1, 0, 1]
                [0, 1, 1, 0]    [1, 0, 0, 0, 1] [1, 1, 1]                                       [1, 1, 1, 1, 1, 1, 1, 1]
                                [0, 1, 1, 1, 1]
Figura [0-6]: 6
```



```
Numero de fila [0-11]: 4
Numero de columna [0-39]: 15
```



```
Opción [0-9]: 6
```



```
Opción [0-9]: 8
El periodo es 30
Opción [0-9]: 0
==============================================
=   Always look on the bright side of life   =
==============================================
```