Proyecto web-prolog

Bancalá Christian – Loza Carlos

Universidad Nacional del Sur

Lógica para ciencias de la computación

**Contenido**

[Funcionalidad básica 2](#_Toc514153422)

[Ayuda Básica 3](#_Toc514153423)

[Algoritmo ayuda Básica: 4](#_Toc514153424)

[Ayuda Adicional 5](#_Toc514153425)

# Funcionalidad básica

sadfsadfasdfasd

# Ayuda Básica

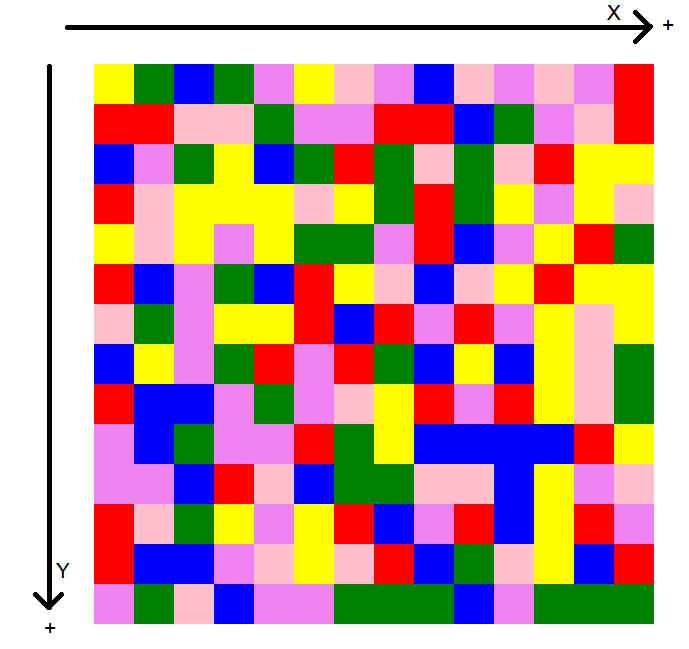
Este método consiste en brindar ayuda al jugador sobre la cantidad de celdas que se agregan a la lista de adyacentes al presionar un cierto color. Particularmente, se muestra en pantalla cuantas celdas se agregan para cada uno de los colores del juego en caso de apretarlo.

Para facilitar la realización del método, se implementó un método auxiliar llamado ‘adyacenteC’ el cual busca la cantidad de celdas adyacentes dado un determinado color y una grilla. Para la resolución de este sub-método se optó por la siguiente estrategia:

Comienza la búsqueda desde la esquina superior izquierda del tablero, recorriendo recursivamente el mismo hacia abajo y hacia la derecha en busca de celdas del mismo color que la celda inicial.   
Si el color de la celda actual, coincide con el de la celda inicial, se agrega a la lista de adyacentes y se continúa con la recursión (siguiendo abajo y a la derecha). De lo contrario, la operación finaliza.

*Puede que una misma celda se agregue dos veces a una lista. Para solucionar esto se implementó una lista sin elementos repetidos.*

De esta manera, se pueden encontrar todas las celdas adyacentes en una grilla con determinada configuración.



En síntesis, la estrategia comienza por la esquina superior izquierda, la cual corresponde a la coordenada (0,0) y, luego continua recursivamente hacia abajo (sumándole 1 a Y) y hacia la derecha (sumándole 1 a X), agregando coordenadas a una lista o finalizando, según corresponda.

## Algoritmo ayuda básica:

1. Utiliza al método ‘adyacenteC’ en busca de la cantidad de adyacentes que hay para el color inicial de la celda (0,0). En el caso del ejemplo, el método comenzaría buscando todos los adyacentes amarillos.
2. Para cada uno de los posibles colores:
   1. Se hace ‘flick’ a la grilla original con el color actual, dando como resultado una nueva grilla modificada (GridN)
   2. Se analiza GridN, nuevamente con ‘adyacenteC.   
      El resultado de esta operación serán las celdas adyacentes del color original, más las nuevas celdas adyacentes que se adicionaron por el cambio de color.
   3. Se retorna en un arreglo el resultado de analizar GridN con ‘adyacenteC’, menos el resultado de ‘adyacenteC’ con la grilla original (esto para cada uno de los colores).

Los resultados se muestran en pantalla mediante código JavaScript.

# Ayuda Adicional

Este método brinda una ayuda extra al jugador y le permite saber cuántas celdas adyacentes se agregarán con cada uno de los colores si primero se elige un color. Es decir, si Ayuda Básica nos brindaba la información de una jugada, este método nos brinda la información de dos jugadas: las celdas adyacentes que se agregan por un primer color elegido más las que se agregarían con cada uno de los otros colores.

Se utilizó nuevamente el método ‘adyacenteC’ detallado anteriormente.

En este caso, primero se averigua la cantidad de celdas adyacentes que se agregan con un determinado color elegido por el usuario, mediante las siguientes líneas de código:

*(getColorEnPos(Grid,0,0,ColorRta),adyacenteC(Grid,(0,0),ColorRta,ConjuntoC)),*

*(flick(Grid,PrimerColor,GridN),adyacenteC(GridN,(0,0),PrimerColor,ConjuntoPC), PrimerColorBeneficio is ConjuntoPC - ConjuntoC),*

En *“PrimerColorBeneficio”* tendremos el resultado de las celdas adyacentes que se agregan en caso de optar por el primer color.

Luego, se hace uso de ayudaBasica, y se averigua cuantas celdas adyacentes se agregan para cada uno de los colores, utilizando la nueva grilla obtenida al hacer flick con el primer color (GridN en el código de arriba).

El resultado será la suma de los resultados del arreglo de AyudaBasica más ‘PrimerColorBeneficio’:

*ayudaBasica(GridN,[Red1,Violet1,Pink1,Green1,Blue1,Yellow1]),*

*Red is Red1 + PrimerColorBeneficio,*

*Violet is Violet1 + PrimerColorBeneficio,*

*Pink is Pink1 + PrimerColorBeneficio,*

*Green is Green1 + PrimerColorBeneficio,*

*Blue is Blue1 + PrimerColorBeneficio,*

*Yellow is Yellow1 + PrimerColorBeneficio.*