INSTITUTO TECNOLÓGICO DE MORELIA

# PROYECTO: Juego batalla naval en Prolog

Programación Lógica y Funcional – Profesor: Ignacio Mota Cruz

Erick Rivas Gómez

27 de mayo de 2019

# PROYECTO: JUEGO BATALLA NAVAL EN PROLOG

## INTRODUCCIÓN

En este proyecto se desarrolla una versión electrónica del juego de batalla naval, bajo el lenguaje de programación Prolog. Se utilizarán reglas, hechos y predicados. Juegan usuario contra PC. El juego permite decidir la dimensión del tablero (cuadrado, por lo que solo se solicita un lado); el número de barcos es el mismo que la dimensión del tablero (lado) y son de dimensión 1; también permite elegir si el usuario juega primero, o la PC lo hace. El juego termina cuando todos los barcos de un jugador son derribados. El juego no permite imprimir el tablero a la consola.

### SWI-PROLOG

SWI-Prolog es una implementación en código abierto (en inglés, open source) del lenguaje de programación Prolog. Su autor principal es Jan Wielemaker. En desarrollo ininterrumpido desde 1987, SWI-Prolog posee un rico conjunto de características, bibliotecas (incluyendo su propia biblioteca para GUI, XPCE), herramientas (incluyendo un IDE) y una documentación extensiva. SWI-Prolog funciona en las plataformas Unix, Windows y Macintosh.

### JUEGO BATALLA NAVAL

La batalla naval (juego de los barquitos o hundir la flota, nombre con el que se comercializó en España el juego de mesa; hundiendo barquitos, en algunos lugares de Hispanoamérica), del nombre en inglés battleship, es un juego tradicional de estrategia y algo de suerte, que involucra a dos participantes.

Se ha comercializado como juego de mesa en distintos formatos por varias marcas. El primero en sacarlo al mercado fue Milton Bradley Company, en 1931, y se jugaba con lápiz y papel. En 2012 se estrenó una película basada en el juego, titulada en inglés Battleship.

## DESARROLLO

El código del archivo puede encontrarse en el repositorio de GitHub:

https://github.com/rivaserick/batalla-naval

Para cargar el archivo, se debe ir a la carpeta donde se encuentra **swiBatallaNaval.pl**, y ejecutar:

```
ruta\del\archivo> swipl swiBatallaNaval.pl
```

- El juego debe comenzar con el mensaje de bienvenida y posteriormente preguntar al usuario la dimensión del tablero.
- Al finalizar el juego (un jugador se queda sin barcos), la ejecución del programa finaliza, y para jugar de nuevo, es necesario volver a ejecutar el comando antes mencionado.

- Un juego de ejemplo se muestra a continuación.

## CAPTURAS DE PANTALLA

A continuación, se muestra el juego ejecutado en la terminal de Windows.

```
Erick Rivas Gomez - 2019
######################################################################
#######                  JUEGO BATALLA NAVAL                  #######
#############                Bienvenidos           ################
######################################################################

Dimension del tablero:: 2.

Dimension: 2

Barcos: 2

2
Coordenada X para barco: |: 0.
Coordenada Y para barco: |: 0.


1
Coordenada X para barco: |: 1.
Coordenada Y para barco: |: 1.


Primer turno para: Usuario(1), PC(Cualquier numero): |: 3.
2
####################################################
                                        TURNO DE PC
                                          1, 1
                                          Â¡Disparaste al barco del usuario!

####################################################
 TURNO DE USUARIO

Ingresa la fila a donde quieres disparar: |: 1.

Ingresa la columna de disparo: |: 0.


Le diste al barco de PC

2
####################################################
                                        TURNO DE PC
                                          0, 0
                                          Â¡Disparaste al barco del usuario!
                                          Â¡Ganaste PC!
```

## CÓDIGO

Código final del juego batalla naval.

**swiBatallaNaval.pl**

```
#!/usr/bin/env swipl swiBatallaNaval.pl

/*
Juego BATALLA NAVAL en SWI-prolog
Erick Rivas Gómez
Programación Lógica y Funcional
Ingeniería en Sistemas Computacionales
Instituto Tecnológico de Morelia
2019

Ejecución en Windows y Linux:

ruta\archivo> swipl swiBatallaNaval.pl
ruta\archivo# swipl -s swiBatallaNaval.pl

Ejecución en SWI-Prolog (desde la ruta del archivo):
?- ['swiBatallaNaval.pl'].

*/


:- dynamic barco/2.
:- dynamic barco_pc/2.

mensaje_de_bienvenida() :-
  write('Erick Rivas Gomez - 2019'),nl,
  write('##########################################################################'),nl,
  write('#######                  JUEGO BATALLA NAVAL               #######'),nl,
  write('#############                Bienvenidos           ###############'),nl,
  write('##########################################################################'),nl.

leer_numero(Mensaje, Numero) :-
  write(Mensaje),
  write(': '),
  read(Numero).

dimension_tablero(Mensaje, Numero) :-
  write(Mensaje),
  write(': '),
  read(Numero),
  assert(dimension(Numero)),
  %retract(dimension(100)),
  nl,
  write('Dimension: '),
  write(Numero),nl.

numero_barcos(Mensaje, Numero) :-
  assert(barcos(Numero)),
  colocar_barcos(Numero).

set_barcos(0).
set_barcos(N) :-
  N>0,
  colocar_barco(),
  %colocar_barco_pc(),
  nl,
  M is N-1,
  set_barcos(M).

colocar_barco() :-
```

```
  write('Coordenada X para barco: '),
  read(CoordX),
  write('Coordenada Y para barco: '),
  read(CoordY),
  assert(barco(CoordX,CoordY)),
  dimension(D),
  write(D),
  random(0, D, CoordXpc),
  write(CoordXpc),
  write(', '),
  random(0, D, CoordYpc),
  assert(barco_pc(CoordXpc, CoordYpc)),
  write(CoordYpc),
  nl.

dispara_usuario() :-
  write('#####################################################'),nl,
  write(' TURNO DE USUARIO'),nl,
  nl,leer_numero('Ingresa la fila a donde quieres disparar', X),
  nl,leer_numero('Ingresa la columna de disparo', Y),nl,nl,
  (
    barco_pc(X, Y) ->
      retract(barco_pc(X, Y)),
      write('Le diste al barco de PC'),
      (
        barco_pc(_, _) ->
          nl;
          write('¡Ganaste!'), nl, halt
      );
      write('Sigue intentando, usuario'), nl, nl
  ).

dispara_pc() :-
  write('#####################################################'),nl,
  write('                                        TURNO DE PC'),nl,
  write('                                          '),
  dimension(D),random(0, D, U), write(U), write(', '),
  dimension(D),random(0, D, V), write(V), nl,
  (
    barco(U, V) ->
      retract(barco(U, V)),
      write('                                          ¡Disparaste al barco del usuario!'),
      nl,
      (
        barco(_, _) ->
          nl;
          write('                                              ¡Ganaste PC!'),
          nl, halt
      );
      write('                                          Sigue intentando, PC'),
      nl, nl
  ).

juego(0).
juego(N, QuienPrimero) :-
  N>0,
  write(N),nl,

  (QuienPrimero == 1 ->
    dispara_usuario(),
    dispara_pc();
    dispara_pc(),
    dispara_usuario()
  ),
  nl,

  %M is N-1,
  juego(N, QuienPrimero).
```

```
barco_para_usuario() :-
  write('Coordenada X para barco: '),
  read(CoordX),
  write('Coordenada Y para barco: '),
  read(CoordY),nl,
  assert(barco(CoordX,CoordY)).

barco_para_pc() :-
  dimension(D),
  random(0, D, CoordXpc),
  %write('                                          '),
  %write(CoordXpc),  // Descomentar estas 3 lineas para poder ver las coordenadas
  %write(', '),      // de los barcos de PC
  random(0, D, CoordYpc),
  %write(CoordYpc),
  (
    barco_pc(CoordXpc, CoordYpc) ->
      barco_para_pc();
      assert(barco_pc(CoordXpc, CoordYpc))
  ).

colocar_barcos(0).
colocar_barcos(N) :-
  N>0,nl,
  write(N),nl,

  barco_para_usuario(),
  barco_para_pc(),

  M is N-1,
  colocar_barcos(M).

:- initialization(main).

main :-
  mensaje_de_bienvenida(),
  nl,
  dimension_tablero('Dimension del tablero:', DimensionTablero),
  nl, write('Barcos: '), write(DimensionTablero),nl,
  numero_barcos('Barcos:', DimensionTablero),nl,
  leer_numero('Primer turno para: Usuario(1), PC(Cualquier numero)', QuienPrimero),
  %AciertosUsuario is 0;
  %AciertosPC is 0;
  juego(DimensionTablero, QuienPrimero).
  main.
```

## CONCLUSIONES

El proyecto se realizó satisfactoriamente, ya que se puede jugar contra la PC y los resultados son muy variados, cualquiera puede ganar y eso hace que sea entretenido. Además, aumenta la complejidad el hecho de no poder ver el tablero de la PC, ni el propio, lo cual lo convierte en una variante muy desafiante de batalla naval.

En este proyecto se aplicaron los conocimientos adquiridos en clase para completar el desarrollo, y se obtuvieron conocimientos adicionales, por ejemplo el uso de números aleatorios, estructuras dinámicas, entre otros, sin los cuales no habría sido posible concluir el proyecto.