INSTITUTO TECNOLÓGICO DE MORELIA

# PROYECTO: Juego de gato en Prolog

Programación Lógica y Funcional – Profesor: Ignacio Mota Cruz

Erick Rivas Gómez 23 de mayo de 2019

# PROYECTO: JUEGO DE GATO EN PROLOG

### INTRODUCCIÓN

En este proyecto se desarrolla una versión electrónica del juego del gato, bajo el lenguaje de programación Prolog. Se utilizarán reglas, hechos y predicados. El juego permite introducir un nombre de jugador, elegir si se juega con la figura "x" u "o", y también si se desea tener el primer o el segundo turno.

### 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 DE GATO

El tres en línea, también conocido como Ceros y Cruces, tres en raya (España, Ecuador y Bolivia), Michi (en Perú), juego del gato, Triqui (en Colombia), Cuadritos, Gato (en Chile, Costa Rica y México), Tatetí (en Argentina), Totito (en Guatemala), Triqui traka, Equis Cero, Tic-Tac-Toe (en Estados Unidos), o la vieja (en Venezuela), es un juego de lápiz y papel entre dos jugadores: O y X, que marcan los espacios de un tablero de 3×3 alternadamente.

### **DESARROLLO**

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

### RUTA\DEL\ARCHIVO> SWIPL GATO.PL

El juego consta de los siguientes métodos para jugar (en ese orden):

- **jugarGato.** Esta sentencia se escribe después de cargar el archivo con swipl, sirve para iniciar el juego y pedir el nombre del jugador. Además, configura más opciones, como la figura con que se jugará y el turno.
- **Tirar(número).** Sirve para hacer una jugada en cierta posición (número) del tablero. Se tiene que repetir hasta que algún jugador gane, o hasta que se acaben los espacios en el tablero.

# CÓDIGO

A continuación, se presenta el código final del juego del gato.

### gato.pl

/\*
Juego de GATO en SWI-prolog
Erick Rivas Gómez
Programación Lógica y Funcional
Ingeniería en Sistemas Computacionales
Instituto Tecnológico de Morelia
2019
\*/
:-dynamic comienza/1. /\*quien comienza 2\*/

```
:-dynamic modo/1. /*estrategia con la que comienza 3*/
comienza(2).
modo(3).
habilitar_gana_o_empata(4).
iguales(X,X,X).
gato: -box(1,X),box(2,Y),box(3,Z),iguales(X,Y,Z).
gato: -box(4,X),box(5,Y),box(6,Z),iguales(X,Y,Z).
gato: -box(7,X),box(8,Y),box(9,Z),iguales(X,Y,Z).
gato:- box(1,X),box(4,Y),box(7,Z),iguales(X,Y,Z).
gato:- box(2,X),box(5,Y),box(8,Z),iguales(X,Y,Z).
gato:- box(3,X),box(6,Y),box(9,Z),iguales(X,Y,Z).
gato:- box(1,X),box(5,Y),box(9,Z),iguales(X,Y,Z).
gato:- box(3,X),box(5,Y),box(7,Z),iguales(X,Y,Z).
centro(5).
opuesta(1,9).
opuesta(9,1).opuesta(2,8).
opuesta(8,2).
opuesta(7,3).opuesta(3,7).
opuesta(4,6).
opuesta(6,4).esquina(1).
esquina(3).
esquina(9).esquina(7).
lateral(2).
lateral(4).lateral(6).
lateral(8).
arista(1,2,3).
arista(3,2,1).
arista(1,4,7).
arista(7,4,1).
arista(3,6,9).
arista(9,6,3).
arista(7,8,9).
arista(9,8,7).
dosIguales(X,X,Z,Z).
dosIguales(X,Z,X,Z).
dosIguales(Z,X,X,Z).
dosIguales(X,X,Z,Z,X).
dosIguales(X,Z,X,Z,X).
dosIguales(Z,X,X,Z,X).
winPlay(Jugador,Jugada):- nb_getval(p,P), nb_getval(h,H), box(1,X),box(2,Y),
        box(3,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada):- nb_getval(p,P), nb_getval(h,H), box(4,X),box(5,Y),
        box(6,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada):- nb_getval(p,P), nb_getval(h,H), box(7,X),box(8,Y),
        box(9,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada):- nb_getval(p,P), nb_getval(h,H), box(1,X),box(4,Y),
        box(7,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada):- nb_getval(p,P), nb_getval(h,H), box(2,X),box(5,Y),
        box(8,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada):- nb_getval(p,P), nb_getval(h,H), box(3,X),box(6,Y),
        box(9,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada): - nb_getval(p,P), nb_getval(h,H), box(1,X),box(5,Y),
        box(9,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
winPlay(Jugador, Jugada): - nb_getval(p,P), nb_getval(h,H), box(3,X),box(5,Y),
        box(7,Z),dosIguales(X,Y,Z,I,J),not(I = P),not(I = H),J = Jugador, Jugada is I.
tablero: - box(1,A),
        box(2,B),box(3,C),box(4,D),
        box(5,E),box(6,F),box(7,G),
        box(8,H),box(9,I),
        write('----'),nl,
        write('| '),write(A),write(' | '),write(B),write(' | '),write(C),write(' |'),nl,
        write(' | '),write(D),write(' | '),write(E),write(' | '),write(F),write(' | '),nl,
```

```
write('-----'),nl,
        write('|'),write(G),write('|'),write(H),write('|'),write(I),write('|'),nl,
        write('
                               '),nl.
jugarGato:- restartGato,write('Hola, vamos a jugar gato, ingresa tu nombre... '),nl,
                         read(Nombre),b_setval(foo,Nombre),nl,b_getval(foo,V),write(V),
                         write(' para jugar, usa :-tirar(numero). Ej.: tirar(5).'),nl,nl,
write('Elije para jugar (1 ==> X), (2 ==> 0)'), read(A),
(A == 1 -> write(1),nb_setval(p,'X'),nb_setval(h,'0');
                                      write(2),nl,nb_setval(p,o),nb_setval(h,x)),
nl,write('La PC juega con '),nb getval(h,S),write(S),nl,nl,
        write('Escribe 1 para empezar y 2 para tirar segundo'), read(B),
        (comienza(1),
         (modo(X);true),
        (retract(modo(_));true),
         ( habilitar_gana_o_empata(Jota),((X+1) < Jota,(X1 is X + 1));(X1 is 1)),</pre>
        assert(modo(X1)),
        (B == 2 -> write('si'), automaticPlay; true)),
        tablero,!.
tirar(X):- nb_getval(p,P),nb_getval(h,H),not(gato),box(X,Y),not(Y = P),not(Y = H),
        retract(box(X,_)), assert(box(X,P)),
        not(gato),(automaticPlay),tablero,!.
tirar(X):-gato->(tablero,nl,write('Ganaste, '),b_getval(foo,V),write(V),nl);
                              not(jugadaPosible),write('Fin de la partida. Empate.'),
                              nl,tablero,!;(write('El casillero está ocupado'),nl,fail).
iaTic(X):- nb getval(p,P),nb getval(h,H),not(gato),box(X,Y),not(Y = P),not(Y = H),
        retract(box(X,_)), assert(box(X,H)),
        not(gato),nl, jugadaPosible,write('Turno de '), nb getval(foo,V),write(V),nl,!.
iaTic(X):-gato->(nl,write('Gana PC, Perdiste, '),b_getval(foo,V),write(V),nl);
                     not(jugadaPosible),nl,fail.
restartGato:-(retract(box(1,_)),retract(box(2,_)),retract(box(3,_)),retract(box(4,_)),
retract(box(5, _)), retract(box(6, _)), retract(box(7, _)), retract(box(8, _)), retract(box(9, _)); true),
(retract(estrategia(_,_));true),
assert(box(1,1)),assert(box(2,2)),assert(box(3,3)),
assert(box(4,4)), assert(box(5,5)), assert(box(6,6)),
assert(box(7,7)), assert(box(8,8)), assert(box(9,9)),
(comienza(Y);true),
(retract(comienza(_));true),
(Y = 1, (Y1 \text{ is } 2); (Y1 \text{ is } 1)),
assert(comienza(Y1)),
!.
automaticPlay:- nb_getval(p,P),nb_getval(h,H),winPlay(H,W)->iaTic(W);
                                  (winPlay(P,M)->iaTic(M)
                                               ;jugadaEstrategica
                                                    ).
jugadaEstrategica:-\ nb\_getval(p,P), nb\_getval(h,H), (estrategia(X,Y),kill\_the\_duck(X,Y));
                     ((not(box(A,H);box(B,P))),primera_jugada);
                     ((not(box(X,H)),box(X,P)),segunda_jugada);
                     (jugadaAleatoria).
primera_jugada:-modo(X),kill_the_duck(X,1).
segunda_jugada:- nb_getval(p,P),box(X,P),
                 (centro(X),kill_the_duck(4,1));
                 (esquina(X),kill_the_duck(5,1));
                 (lateral(X), kill the duck(6,1,X))
```

```
kill_the_duck(1,1):- (retract(estrategia(_,_));true),
                     centro(X),iaTic(X),
                    assert(estrategia(1,2)).
kill_the_duck(1,2):- (retract(estrategia(_,_));true),
                     esquina(X),iaTic(X),
                assert(estrategia(1,3)).
kill_the_duck(1,3):- nb_getval(p,P),nb_getval(h,H),(retract(estrategia(_,_));true),
                    esquina(X),
                    arista(X,Y,Z),
                    box(Y,Y1),
                    not(Y1=P),
                    box(Z,Z1),
                    Z1=H,
                    iaTic(X).
kill_the_duck(2,1):-(retract(estrategia(_,_));true),
                    esquina(X),iaTic(X),
                    assert(estrategia(2,2)),
                    assert(last_move(X)).
kill_the_duck(2,2):-(retract(estrategia(_,_));true),
                    last_move(Y),
                     (retract(last_move(_));true),
                    opuesta(Y,X), iaTic(X),
                    assert(estrategia(2,3)).
kill the duck(2,3):-(retract(estrategia( , ));true),
                    esquina(X),iaTic(X).
kill_the_duck(3,1):-(retract(estrategia(_,_));true),
                    esquina(X),iaTic(X),
                    assert(estrategia(3,2)),
                    assert(last move(X)).
kill_the_duck(3,2):-(retract(estrategia(_,_));true),
                    last_move(Y),
                     (retract(last move( ));true),
                     (centro(C), lateral(X), arista(Y, X, Z),
                    jugadaPosible(C),
                    jugadaPosible(X),
                     jugadaPosible(Z),iaTic(X),
                    assert(estrategia(3,3)));
                    opuesta(Y,X),iaTic(X),
                    assert(estrategia(2,3)).
kill_the_duck(3,3):-(retract(estrategia(_,_));true),
                    centro(X),iaTic(X).
kill_the_duck(4,1):-esquina(Y),
                    iaTic(Y),
                    (retract(estrategia(_,_));true),
                    assert(estrategia(4,2)).
kill_the_duck(4,2):-retract(estrategia(4,2)),(esquina(Y),iaTic(Y)).
kill_the_duck(5,1):-(retract(estrategia(_,_));true),
                     (centro(Y),iaTic(Y)),
                    assert(estrategia(5,2)).
kill_the_duck(5,2):-retract(estrategia(5,2)),
                     (lateral(Y), iaTic(Y)).
kill_the_duck(6,1,X):-(opuesta(X,Y),iaTic(Y)).
```

```
jugadaAleatoria:-centro(X),iaTic(X),!.
jugadaAleatoria:-nb_getval(p,P),nb_getval(h,H),box(X,Y),not(Y=P;Y=H),iaTic(X),!.
jugadaPosible:-nb_getval(p,P),nb_getval(h,H),box(X,Y),not(Y=P;Y=H).
jugadaPosible(X):-nb_getval(p,P),nb_getval(h,H),box(X,Y),not(Y=P;Y=H).
```

### CAPTURAS DE PANTALLA

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

```
\swi-prolog>swipl gato.pl
                                                                                       ------
Welcome to SWI-Prolog (threaded, 64 bits, version 8.0.2)
                                                                                       true.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
                                                                                       3 ?- tirar(9).
Please run ?- license. for legal details.
For online help and background, visit http://www.swi-prolog.org
                                                                                       Turno de []
For built-in help, use ?- help(Topic). or ?- apropos(Word).
                                                                                       0 2 X
1 ?- jugarGato.
Hola, vamos a jugar gato, ingresa tu nombre...
                                                                                       |4|0|0|
: erick.
                                                                                       7 8 X
erick para jugar, usa :-tirar(numero). Ej.: tirar(5).
                                                                                       true.
Elije para jugar (1 ==> X), (2 ==> 0)|: 1.
                                                                                       4 ?- tirar(4).
La PC juega con O
                                                                                       Turno de []
Escribe 1 para empezar y 2 para tirar segundo: 2.
                                                                                       0 0 X
Turno de erick
                                                                                       | X | 0 | 0 |
1 1 2 | 3 |
                                                                                       7 8 X
|4|0|6|
                                                                                       true.
7 8 9
                                                                                       5 ?- tirar(8).
true.
                                                                                       Fin de la partida. Empate.
2 ?- tirar(3).
                                                                                       0 0 X
Turno de []
                                                                                       | X | 0 | 0 |
| 0 | 2 | X |
                                                                                       | 0 | X | X |
14 0 6
                                                                                       true.
7 | 8 | 9 |
                                                                                       6 2- 1
```

## 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.

Por otra parte, al utilizar este lenguaje de programación, se logra que las operaciones que en otros lenguajes serían muy complejas, se realicen más fácilmente e incluso ahorrando líneas de código. Prolog es un lenguaje muy potente aplicado a problemas que se puedan resolver de manera lógica, por lo que creo que es muy útil aprenderlo para poder aplicarlo en la resolución de problemas en el futuro.