Rentaro Tsunamura

Final: Prolog

What is the Prolog?

Prolog is a programming language used to solve problems about objects and relations between them. The language has the advantage of allowing programs to represent themselves declaratively, and Prolog is based on predicate logic, unlike other languages, which take a procedural approach. Prolog is useful in the field of artificial intelligence for natural language analysis, comprehension, and inference. It was developed by Alain Colmerauer around 1970 and it was based on the Q-System for natural language analysis.

Structure of Prolog

A Prolog program consists of three elements.

**Facts** - A fact in Prolog is to declare some truth about an entity and its relations.

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

[Prolog Examples (csus.edu)](https://athena.ecs.csus.edu/~mei/logicp/prolog/programming-examples.html)

**Rules** - A rule in Prolog is to define a condition or principle about an entity and its relations.

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

[Prolog Examples (csus.edu)](https://athena.ecs.csus.edu/~mei/logicp/prolog/programming-examples.html)

**Queries** - Queries in Prolog are asking questions about entities and its relationships.

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

[Prolog Examples (csus.edu)](https://athena.ecs.csus.edu/~mei/logicp/prolog/programming-examples.html)

Why I Pick Up Prolog?

The reason why I chose Prolog is because the name is more attractive than the other two languages. In addition, I thought it is interesting to implement it in board games such as chess, because it can define logical rules for strategy and find the next best step.

My Example Code

Environment

The environment I developed this time is visual studio code. This time I used tau-prolog to write the prolog code in html. Tau-Prolog is a Prolog implementation written in JavaScript and used primarily in a web environment. It supports asynchronous processing and allows interactive programming. It is open source and can be downloaded here: [Tau Prolog: Downloads (tau-prolog.org)](http://tau-prolog.org/downloads) .

Submarine.html

<!DOCTYPE html>

<html>

    <head>

        <title>Deep submarine simulation</title>

        <meta charset="utf-8">

        <link href="submarine.css" rel="stylesheet">

        <script type="text/javascript" src="modules/tau-prolog.js"></script>

        <script type="text/prolog" id="submarine.pl">

        :- use\_module(library(dom)).

% move/3

% Move the submarine

move(Submarine, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Submarine, Axis, px(X0)),

    X1 is X0+Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1220),

    style(Submarine, Axis, px(X1)).

% mov/3 for lanternfish

moveL(Lanternfish, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Lanternfish, Axis, px(X0)),

    X1 is X0 + Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1220),

    style(Lanternfish, Axis, px(X1)).

% mov/3 for macropinna

moveM(Macropinna, Axis, Incr) :-

        get\_by\_id(board, Board),

        style(Macropinna, Axis, px(X0)),

        X1 is X0 + Incr,

        X1 >= -10,

        (Axis = top -> X1 =< 640 ; X1 =< 1220),

        style(Macropinna, Axis, px(X1)).

% moveT/3 for tuna

moveT(Tuna, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Tuna, Axis, px(X0)),

    X1 is X0 + Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1200),

    style(Tuna, Axis, px(X1)).

% mov/3 for dolphin

moveD(Dolphin, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Dolphin, Axis, px(X0)),

    X1 is X0 + Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1140),

    style(Dolphin, Axis, px(X1)).

% mov/3 for octopus

moveO(Octopus, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Octopus, Axis, px(X0)),

    X1 is X0 + Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1220),

    style(Octopus, Axis, px(X1)).

% mov/3 for snapper

moveS(Snapper, Axis, Incr) :-

    get\_by\_id(board, Board),

    style(Snapper, Axis, px(X0)),

    X1 is X0 + Incr,

    X1 >= -10,

    (Axis = top -> X1 =< 640 ; X1 =< 1220),

    style(Snapper, Axis, px(X1)).

% anim/2

% Change animation of the submarine

anim(Submarine, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Submarine, backgroundImage, url(Src)).

% anim/2 for lanternfish

animL(Lanternfish, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Lanternfish, backgroundImage, url(Src)).

% anim/2 for macropinna

animM(Macropinna, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Macropinna, backgroundImage, url(Src)).

% anim/2 for tuna

animT(Tuna, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Tuna, backgroundImage, url(Src)).

% anim/2 for dolphin

animD(Dolphin, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Dolphin, backgroundImage, url(Src)).

% anim/2 for octopus

animO(Octopus, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Octopus, backgroundImage, url(Src)).

% anim/2 for snapper

animS(Snapper, Dir) :-

    atom\_concat('res/', Dir, Src1),

    atom\_concat(Src1, '.png', Src),

    style(Snapper, backgroundImage, url(Src)).

% remark\_control/1

% Remark the pressed control

remark\_control(Key) :-

    clear\_controls,

    atom\_concat('control-', Key, Id),

    get\_by\_id(Id, Control),

    add\_class(Control, focus).

% clear\_control/0

% Remove the focus class of any control

clear\_controls :-

    findall(X, (get\_by\_class(control, X), remove\_class(X, focus)), \_).

% action/2

% up

action(Submarine,Lanternfish,Macropinna,Tuna, Dolphin,Octopus, Snapper, w) :-

    anim(Submarine, up),

    anim(Lanternfish, lanternfish\_right),

    anim(Macropinna, macropinna\_left),

    anim(Octopus,octopus\_right),

    anim(Snapper, snapper\_left),

    anim(Tuna, tuna\_left),

    anim(Dolphin, dolphin\_left),

    move(Submarine, top, -5),

    move(Lanternfish, left, 1),

    move(Macropinna, left, -1),

    move(Octopus, left, 1),

    move(Snapper, left, -3),

    move(Tuna, left, -4),

    move(Dolphin, left, -5).

% down

action(Submarine, Lanternfish, Macropinna,Tuna, Dolphin, Octopus,Snapper, s) :-

    anim(Submarine, down),

    anim(Lanternfish, lanternfish\_left),

    anim(Macropinna, macropinna\_right),

    anim(Octopus,octopus\_left),

    anim(Snapper, snapper\_left),

    anim(Tuna, tuna\_left),

    anim(Dolphin, dolphin\_right),

    move(Submarine, top,  5),

    move(Lanternfish, left, -1),

    move(Macropinna, left, 1),

    move(Octopus, left, -1),

    move(Snapper, left, -3),

    move(Tuna, left, -4),

    move(Dolphin, left, 5).

% left

action(Submarine,Lanternfish,Macropinna,Tuna,Dolphin, Octopus,Snapper, a) :-

    anim(Submarine, left),

    anim(Lanternfish, lanternfish\_right),

    anim(Macropinna, macropinna\_left),

    anim(Octopus,octopus\_top),

    anim(Snapper,snapper\_right),

    anim(Tuna, tuna\_right),

    anim(Dolphin, dolphin\_left),

    move(Submarine, left, -5),

    move(Lanternfish, left, 1),

    move(Macropinna, left, -1),

    move(Octopus, left, 1),

    move(Snapper, left, 3),

    move(Tuna, left, 4),

    move(Dolphin, left, -5).

% right

action(Submarine,Lanternfish,Macropinna,Tuna,Dolphin,Octopus,Snapper, d) :-

    anim(Submarine, right),

    anim(Lanternfish, lanternfish\_left),

    anim(Macropinna, macropinna\_right),

    anim(Octopus,octopus\_down),

    anim(Snapper, snapper\_right),

    anim(Tuna, tuna\_right),

    anim(Dolphin, dolphin\_right),

    move(Submarine, left, 5),

    move(Lanternfish, left, -1),

    move(Macropinna, left, 1),

    move(Octopus, left, -1),

    move(Snapper, left, 3),

    move(Tuna, left, 4),

    move(Dolphin, left, 5).

% bubble

action(Submarine,Lanternfish, Macropinna, Tuna, Dolphin, Octopus,Snapper, p) :-

    style(Submarine, top, px(Y0)), style(Submarine, left, px(X0)),

    Y1 is Y0+50, X1 is X0+37,

    create(div, Bubble),

    style(Bubble, top, px(Y1)), style(Bubble, left, px(X1)),

    add\_class(Bubble, bubble),

    insert\_before(Bubble, Submarine).

% init/0

% Initilize the game

init :-

    get\_by\_id(submarine, Submarine),

    get\_by\_tag(body, Body),

    get\_by\_id(lanternfish, Lanternfish),

    get\_by\_id(macropinna, Macropinna),

    get\_by\_id(tuna, Tuna),

    get\_by\_id(dolphin, Dolphin),

    get\_by\_id(octopus, Octopus),

    get\_by\_id(snapper, Snapper),

    bind(Body, keyup, \_, clear\_controls),

    bind(Body, keydown, Event, (

        event\_property(Event, key, Key),

        remark\_control(Key),

        action(Submarine,Lanternfish, Macropinna, Tuna, Dolphin, Octopus,Snapper, Key)

    )).

        </script>

    </head>

    <body>

        <!-- Game board container -->

        <div id="board">

            <div id="lanternfish"></div>

            <div id="macropinna"></div>

            <div id="tuna"></div>

            <div id="dolphin"></div>

            <div id="octopus"></div>

            <div id="snapper"></div>

            <!-- submarine element representing the game character -->

            <div id="submarine"></div>

            <!-- Control buttons for game actions -->

            <div id="controls">

                <div class="control" id="control-w">w</div>

                <div class="control" id="control-a">a</div>

                <div class="control" id="control-s">s</div>

                <div class="control" id="control-d">d</div>

                <div class="control" id="control-p">p (bubble)</div>

            </div>

        </div>

        <!-- JavaScript code for initializing the game -->

        <script type="text/javascript">

            // Create Tau Prolog session with a timeout of 1000 milliseconds

            var session = pl.create(1000);

            // Consult the Prolog program file "submarine.pl"

            session.consult("submarine.pl", {

                success: function() {

                    // Query the Prolog goal "init."

                    session.query("init.", {

                        success: function() {

                            // Find answers and initiate the game logic

                            session.answer();

                        }

                    });

                }

            });

        </script>

    </body>

</html>

Code Description

コンピューターのスクリーンショット

自動的に生成された説明

First, get\_by\_id and get\_by\_tag is used to assign specific elements in the HTML to Prolog variables. The bind is used to set up processing for keydown and keyup keyboard events. The processing is specified when a key is pressed and when a key is released.

コンピューターのスクリーンショット

自動的に生成された説明

It defines the action when the player presses the "W" key (up). The anim() sets the animation of each object. For example, anim(Submarine, up) changes the animation of a submarine to face upward. The move() moves each object a specified number of pixels in a specified direction. For example, move(Submarine, top, -5) moves the submarine up -5 pixels.

コンピューターのスクリーンショット

自動的に生成された説明

The logic is defined when the submarine moves a specified number of pixels on a specified axis.

コンピューターのスクリーンショット

自動的に生成された説明

Logic is defined to change the submarine's animation based on the specified direction (Dir) and display the image in the new direction.

Modification

I made some modifications to the example code to make it more interesting. First, I increased the number of objects from one to seven. The main object is a submarine, and the other objects are different kinds of fish. The submarine moves in the direction of the user's keyboard input, but the other fish move in random directions I determine. The distance moved also depends on the fish. Faster fish move farther, slower fish move less.

How to execute the program

Paste the submarine.html file on the web.

Output

コンピューターの画面

自動的に生成された説明

Press W to move the submarine up.

Press A to move left.

Press D to move right.

Press S to move down.

Press P to display the bubble.

Note that running it for a long time will slow down the processing speed.

Prolog vs Other Language

Compared to other languages, Prolog is able to express logical game logics, allowing it to avoid complex processing. Also, Readability and Writability may be better than other languages because of the simplicity of the code.

Summary

I did not have much time for this assignment, so I only understood a little about Prolog, but I thought it was a very interesting language. I would also like to learn a little more about prolog.