CSC 4301, Project #2

# Logical agent for the Wumpus game.

Pr. Tajjedine Rachidi

Ziad El Ismaili

Abdelhamid El Hand

Imad-Eddine Ouahidi

Al Akhawayn University in Ifrane

Saturday, June 11, 2022

## *Table of Contents*

## ***I. Introduction:***

In this project, we will be implementing a logical agent which if asked about one single action in one single iteration, such as safety of rooms, shooting the Wumpus or picking up gold, it would answer by true, false or values of rooms. First, we will be implementing the Wumpus game environment, using a 4x4 grid which we will be displaying, (x) as the current position of the agent, (+) for the already visited room, and (-) the empty rooms.

## ***II. Key predicates and meaning of variables:***

### ***Useful predicates:***

We start by typing run(N)., where N is the world number in the worldbuilder.pl file. The above predicates are used to initialize the game. Moreover, we used some predicates to draw the grid, using lists implementation.

```
run(N) :-
   format('Welcome to Wumpus World~n'),
   create_matrix(4,4,M),
   replace_row_col(M,1,1,x,NewMatrix),
   assert(map(NewMatrix)),
   display_map(NewMatrix),
   init(N).

init(N) :-
   init_agent,
   init_states_wumpus(N),
   init_sensors.

init_sensors :-
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).

init_states_wumpus(N) :-
   consult('worldBuilder.pl'),
   buildWorld(N).
```

```
init_agent :-
    assert(agent_position(1,1)),
    assert(agent_health(alive)),
    assert(agent_in_cave(yes)),
    assert(agent_arrows(1)),
    assert(gold(0)).

% Creating the gird

do_list(N,L) :- do_list1(N, [], L).
do_list1(0, L, L) :- !.
do_list1(N, R, L) :-
    N > 0,
    N1 is N-1,
    do_list1(N1, [-|R], L).

create_matrix(Rows,Cols,Matrix) :-
    do_list(Cols,List),
    create_matrix1(Rows,[],List,Matrix).


create_matrix1(0,M,_,M).
create_matrix1(N,R,L,M) :-
    N > 0,
    N1 is N - 1,
    append(R,[L],L1),
    create_matrix1(N1, L1,L, M).

replace_nth(Index,List,Value,NewList) :-
    nth1(Index,List,_,Transfer),
    nth1(Index,NewList,Value,Transfer).

replace_row_col(M,Row,Col,Cell,N) :-
    nth1(Row,M,Old),
    replace_nth(Col,Old,Cell,Upd),
    replace_nth(Row,M,Upd,N).

write_matrix([]).
write_matrix([H|T]) :-
    write_row(H),
    write_matrix(T).

write_row([]) :-
    format('|\n').
write_row([H|T]) :-
    format('| ~p ',H),
    write_row(T).
```

Next, we used the following predicates to implement the movement of the agent, which will be up, down, right, left. The user will be typing the following predicates and the map of the Wumpus world will update accordingly:

```
% ------- Moves of the agent --------
up :-
   agent_position(X,Y),
   X1 is X - 1,
   map(M),
   replace_row_col(M,X,Y,+,M1),
   replace_row_col(M1,X1,Y,x,NewMatrix),
   display_map(NewMatrix),
   format('PosAgent: ~p\n',[agent_position(X1,Y)]),
   % Update Position of Agent and Map
   retractall(agent_position(,)),
   assert(agent_position(X1,Y)),
   retractall(map(_)),
   assert(map(NewMatrix)),
   update_agent_health,
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).

down :-
   agent_position(X,Y),
   X1 is X + 1,
   map(M),
   replace_row_col(M,X,Y,+,M1),
   replace_row_col(M1,X1,Y,x,NewMatrix),
   display_map(NewMatrix),
   format('Accion: ~p\n',[agent_position(X1,Y)]),
   % Update Position of Agent and Map
   retractall(agent_position(,)),
   assert(agent_position(X1,Y)),
   retractall(map(_)),
   assert(map(NewMatrix)),
   update_agent_health,
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).
```

```
right :-
   agent_position(X,Y),
   Y1 is Y + 1,
   map(M),
   replace_row_col(M,X,Y,+,M1),
   replace_row_col(M1,X,Y1,x,NewMatrix),
   display_map(NewMatrix),
   format('Accion: ~p\n',[agent_position(X,Y1)]),
   % Update Position of Agent and Map
   retractall(agent_position(,)),
   assert(agent_position(X,Y1)),
   retractall(map(_)),
   assert(map(NewMatrix)),
   update_agent_health,
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).

left :-
   agent_position(X,Y),
   Y1 is Y - 1,
   map(M),
   replace_row_col(M,X,Y,+,M1),
   replace_row_col(M1,X,Y1,x,NewMatrix),
   display_map(NewMatrix),
   format('Accion: ~p\n',[agent_position(X,Y1)]),
   % Update Position of Agent and Map
   retractall(agent_position(,)),
   assert(agent_position(X,Y1)),
   retractall(map(_)),
   assert(map(NewMatrix)),
   update_agent_health,
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).
```

The predicate agent-position(X,Y) will return the agent current position, and all of the code inside is concerned with updating the map and the sensors of the agent.

After that, we designed some predicates to handle the actions of the agent, such as grabbing the gold, climbing, and shooting the Wumpus.

```
% Actions of Agent
grab :-
   pickup_gold,
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(no)]\n', [Stench,Breeze,Glitter]).

shoot :-
   stench(Stench),
   breeze(Breeze),
   glitter(Glitter),
   shoot_arrow(Scream),
   format('Sensors: [Stench(~p),Breeze(~p),Glitter(~p),Scream(~p)]\n', [Stench,Breeze,Glitter,Scream]).

climb :-
   can_leave_cave(Answer),
   Answer \== yes, !.

can_leave_cave(yes) :-
   agent_position(1,1), !,
   retract(agent_in_cave(yes)),
   assert(agent_in_cave(no)),
   format('You are out of cave\n').

can_leave_cave(no) :-
   format("You cannot leave the cave from here.~n").

kill_wumpus(X,Y) :-
   retract(wumpus_position(X,Y)).

update_arrows :-
   agent_arrows(Arrows),
   Arrows > 0, !,
   NewArrows is Arrows - 1,
   retract(agent_arrows(Arrows)),
   assert(agent_arrows(NewArrows)),
   format("You have ~d arrow(s).~n",NewArrows).

shoot_arrow(Scream) :-
   format('Select a direction to shoot:
         up
         down
         right
         left \n'),
   read(ArrowDirection),
   agent_position(X,Y),
   propagate_arrow(X,Y,ArrowDirection,Scream),
   update_arrows.

shoot_arrow(no) :-
   write('No arrows\n').
```

When we select shoot, the user will be prompt which direction to shoot, if the Wumpus is then killed we will be hearing a scream.

***Key predicates:***

- stench() predicate is initially stench(no), but the following conditions will check for the Wumpus in the adjacent rooms, and return true or stench(yes) if it is.

```
stench(yes) :-
  agent_position(X,Y),
  X1 is X + 1,
  X0 is X - 1,
  Y1 is Y + 1,
  Y0 is Y - 1,
 ( wumpus_position(X1,Y) ;
  wumpus_position(X0,Y) ;
  wumpus_position(X,Y1) ;
  wumpus_position(X,Y0) ;
  wumpus_position(X,Y) ),
 !.
```

- breeze() predicate is initially breeze(no), but the following conditions will check for a Pit in the adjacent rooms, and return true or breeze(yes) if it is.

```
-     breeze(yes) :-
-       agent_position(X,Y),
-       X1 is X + 1,
-       X0 is X - 1,
-       Y1 is Y + 1,
-       Y0 is Y - 1,
-       ( pit_position(X1,Y) ;
-         pit_position(X0,Y) ;
-         pit_position(X,Y1) ;
-         pit_position(X,Y0) ;
-         pit_position(X,Y)  ),
-       !.
```

- glitter() predicate is initially glitter(no), but the following conditions will check if the gold is in the current position and return true or glitter(yes) if it is.

```
-   glitter(yes) :-
-     agent_position(X,Y),
-     gold_position(X,Y),
-     !.
```

- pickup_gold() will return true and pick the gold if it is in the current position, or false if it is not. It will also print how many gold the user have if the world has more than one.

```
-   pickup_gold :-
-       agent_position(X,Y),
-       gold_position(X,Y), !,
-       gold(NumberGold),
-       NumberGold1 is NumberGold + 1,
-       retract(gold(NumberGold)),
-       assert(gold(NumberGold1)),
-       format("You have ~d piece(s) of gold!~n",NumberGold1),
-       retract(gold_position(X,Y)).
```

- The predicate grabGold() will return true if the gold is in the current position and print its location, or false if it is not.

```
%Returns true if gold is in current position
grabGold() :-
  agent_position(X,Y),
  gold_position(X,Y),
  format('Gold found in: (~d,~d), PICK IT!\n', [X,Y]),
  !.
```

- The 4 following predicates *check_down_room(), check_up_room(), check_right_room(), check_left_room(),* will check if the adjacent room of the current position are safe. The predicates will print all the safe rooms. Here, we consider X as the vertical axis and Y as the horizontal axis. Thus, for instance when we say. X1 is X + 1, it means we are moving to check the bottom room, since we are increasing in X. We are using < 5 or > 0 to no check the rooms out of the map.

```
-    %Returns true if  the bottom room (X,Y) is safe, meaning contains no pit or wumpus.
-    check_down_room() :-
-        agent_position(X,Y),
-        X1 is X + 1,
-        X1 < 5,
-      \+ pit_position(X1,Y), \+ wumpus_position(X1,Y) -> format('Room (~d,~d) is safe.\n', [X1,Y]).
-
-    %Returns true if the top room (X,Y) is safe, meaning contains no pit or wumpus.
-    check_up_room() :-
-        agent_position(X,Y),
-        X1 is X - 1,
-        X1 > 0,
-      \+ pit_position(X1,Y), \+ wumpus_position(X1,Y) -> format('Room (~d,~d) is safe.\n', [X1,Y]).
-
-    %Returns true if the right room (X,Y) is safe, meaning contains no pit or wumpus.
-    check_right_room() :-
-        agent_position(X,Y),
-        Y1 is Y + 1,
-        Y1 < 5,
-        \+ pit_position(X,Y1), \+ wumpus_position(X,Y1) -> format('Room (~d,~d) is safe.\n', [X,Y1]).
-
-    %Returns true if the left room (X,Y) is safe, meaning contains no pit or wumpus.
-    check_left_room() :-
-        agent_position(X,Y),
-        Y1 is Y - 1,
-        Y1 > 0,
-      \+ pit_position(X,Y1), \+ wumpus_position(X,Y1) -> format('Room (~d,~d) is safe.\n', [X,Y1]).
-
```

- safe() will call the 4 previous predicates above, then, the safe rooms that do not contain a Wumpus, a Pit and that are inside the world will be printed.

```
%Returns all the adjacent rooms that are safe.
safe() :-
   check_left_room();
   check_right_room();
   check_up_room();
   check_down_room(),!.
```

- shootWumpus() will return true if the Wumpus is on one of the adjacent rooms, and false if it is not.

```
-    %Returns true if the wumpus is on an adjacent rooms.
-    shootWumpus() :-
-        agent_position(X,Y),
-        X1 is X + 1,
-        X0 is X - 1,
-        Y1 is Y + 1,
-        Y0 is Y - 1,
-        ( wumpus_position(X1,Y) ;
-            wumpus_position(X0,Y) ;
-            wumpus_position(X,Y1) ;
-            wumpus_position(X,Y0) ;
-            wumpus_position(X,Y)  ),
-        !.
```

Moreover, we can use the following predicates, (usually if we are not playing) to know the locations of the pit, Wumpus, and the gold in the world N.

- wumpus_position(X,Y), where X,Y are the locations inputted by the user, and it will return true if the wumpus is at the location (X,Y) or false if it is not.
- pit_position(X,Y), where X,Y are the locations inputted by the user, and it will return true if a pit is at the location (X,Y) or false if it is not.
- gold_position(X,Y), where X, Y are the locations inputted by the user, and it will return true if the gold is in location (X,Y) or false if it is not.

## ***III. Code testing:***

***N.B:*** *You might need to be running the file main.pl on Swi-prolog version 8.4.2 if you find some issues or compilation errors while running the code.*

In this part, we are going to execute the world N = 1 found in worldbuilder.pl. These are the following configuration of the world:

```
buildWorld(1) :-
    asserta(gold_position(2,3)),
    asserta(wumpus_position(1,3)),
    asserta(pit_position(3,1)),
    asserta(pit_position(3,3)),
    asserta(pit_position(4,4)).
```

The algorithm would be the following:

1. Check if there is gold using grabGold() or if Glitter(yes).

   If true, use *grab* to grab it.

2. Check which rooms are safe using safe().

   Move to one of the safe rooms, using *up, down, right, left*.

3. Check if there is Wumpus on one of the adjacent rooms using shootWumpus() or if Stench(yes).

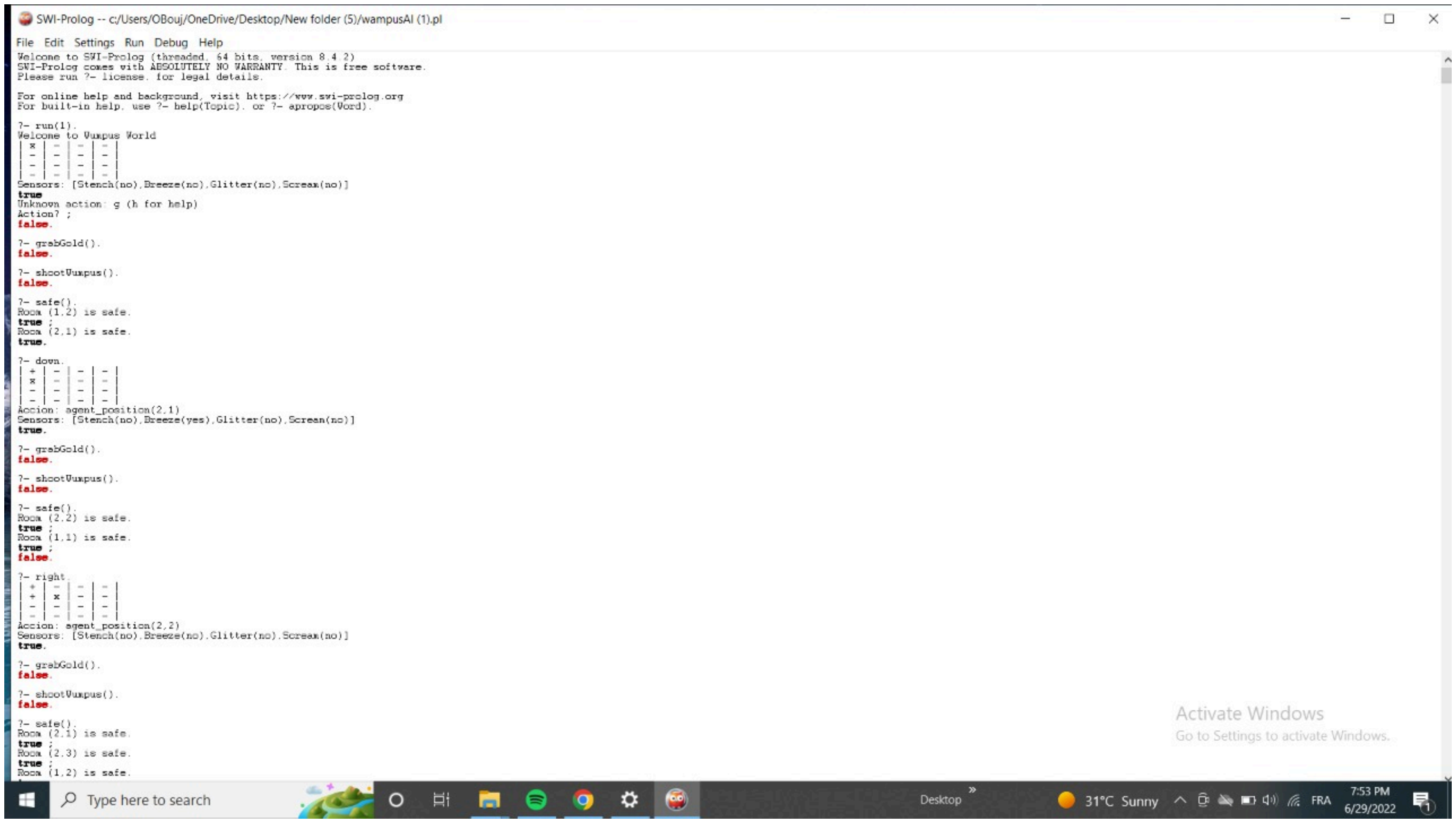
   If true, the user will try to shoot the Wumpus using *shoot* or move to a safe room.

4. If Wumpus is killed and gold is grabbed, move to location (1,1).

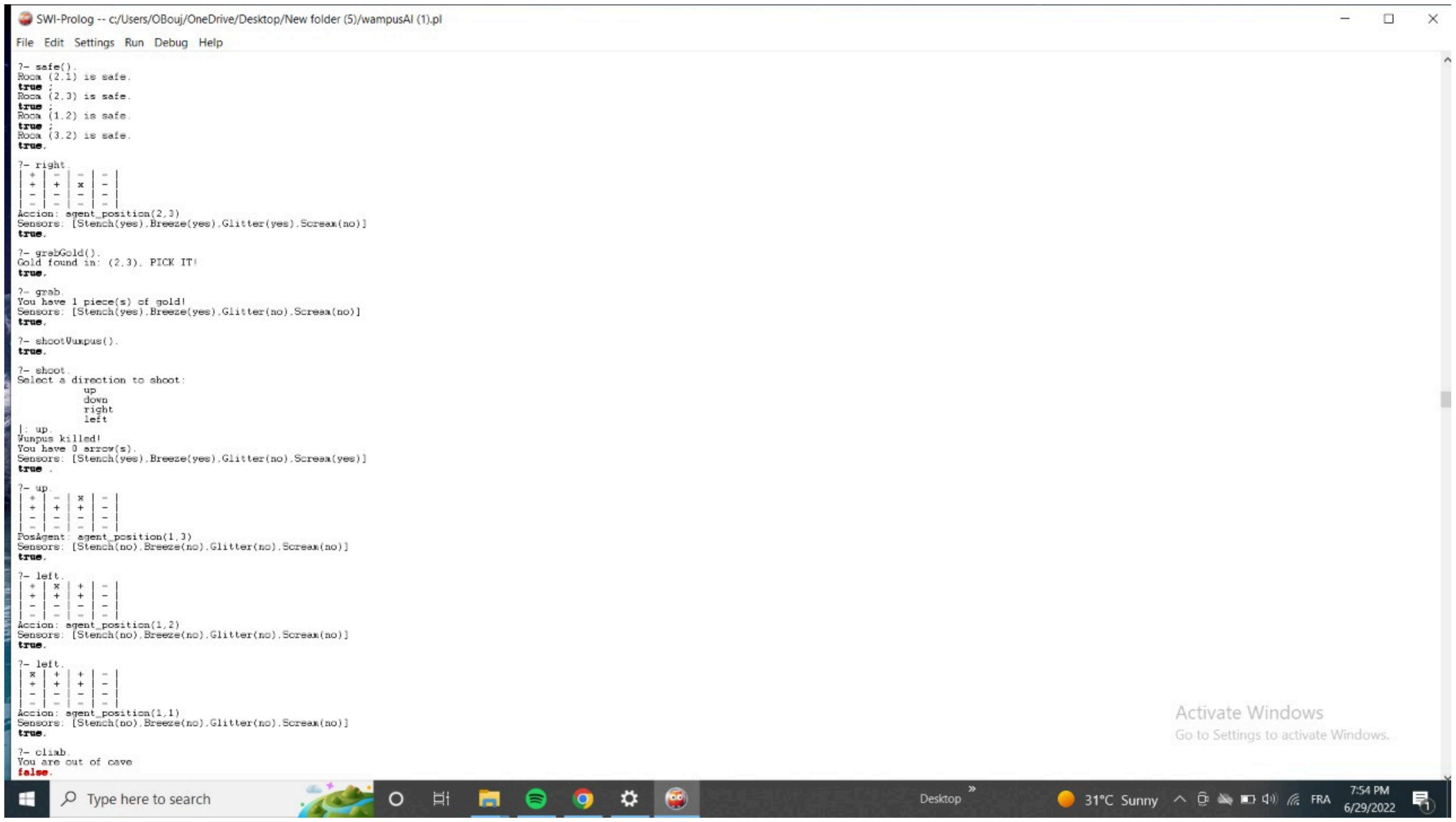
   If agent is in location (1,1) and wants to leave, then use *climb* to leave the cave.

***Screenshots of the execution of the world N = 1:***

The following are screenshots of the previous predicates implemented in the Wumpus game of with world configuration N =1. *(N.B: We can use shootWumpus(), or grabGold() only after Stench(Yes) and Glitter(yes), to save time.)*


SWI-Prolog -- c:/Users/OBouj/OneDrive/Desktop/New folder (5)/wampusAI (1).pl
File Edit Settings Run Debug Help
Activate Windows
Go to Settings to activate Windows.
Type here to search
Desktop
31°C Sunny
FRA
7:53 PM
6/29/2022

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- run(1).
Welcome to Wumpus World
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true
Unknown action: g (h for help)
Action? ;
false.

?- grabGold().
false.

?- shootWumpus().
false.

?- safe().
Room (1,2) is safe.
true ;
Room (2,1) is safe.
true.

?- down.
| + | - | - | - |
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,1)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- grabGold().
false.

?- shootWumpus().
false.

?- safe().
Room (2,2) is safe.
true ;
Room (1,1) is safe.
true ;
false.

?- right.
| + | - | - | - |
| + | x | - | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,2)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- grabGold().
false.

?- shootWumpus().
false.

?- safe().
Room (2,1) is safe.
true ;
Room (2,3) is safe.
true ;
Room (1,2) is safe.
```


SWI-Prolog -- c:/Users/OBouj/OneDrive/Desktop/New folder (5)/wampusAI (1).pl
File Edit Settings Run Debug Help
Activate Windows
Go to Settings to activate Windows.
Type here to search
Desktop
31°C Sunny
FRA
7:54 PM
6/29/2022

```
?- safe().
Room (2,1) is safe.
true ;
Room (2,3) is safe.
true ;
Room (1,2) is safe.
true ;
Room (3,2) is safe.
true.

?- right.
| + | - | - | - |
| + | + | x | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,3)
Sensors: [Stench(yes),Breeze(yes),Glitter(yes),Scream(no)]
true.

?- grabGold().
Gold found in: (2,3). PICK IT!
true.

?- grab.
You have 1 piece(s) of gold!
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(no)]
true.

?- shootWumpus().
true.

?- shoot.
Select a direction to shoot:
           up
           down
           right
           left
|: up.
Wumpus killed!
You have 0 arrow(s).
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(yes)]
true .

?- up.
| + | - | x | - |
| + | + | + | - |
| - | - | - | - |
| - | - | - | - |
PosAgent: agent_position(1,3)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- left.
| + | x | + | - |
| + | + | + | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(1,2)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- left.
| x | + | + | - |
| + | + | + | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(1,1)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- climb.
You are out of cave
false.
```

Now, we are going to try playing as a user on random different worlds, then put the results of some of the worlds in the following table:

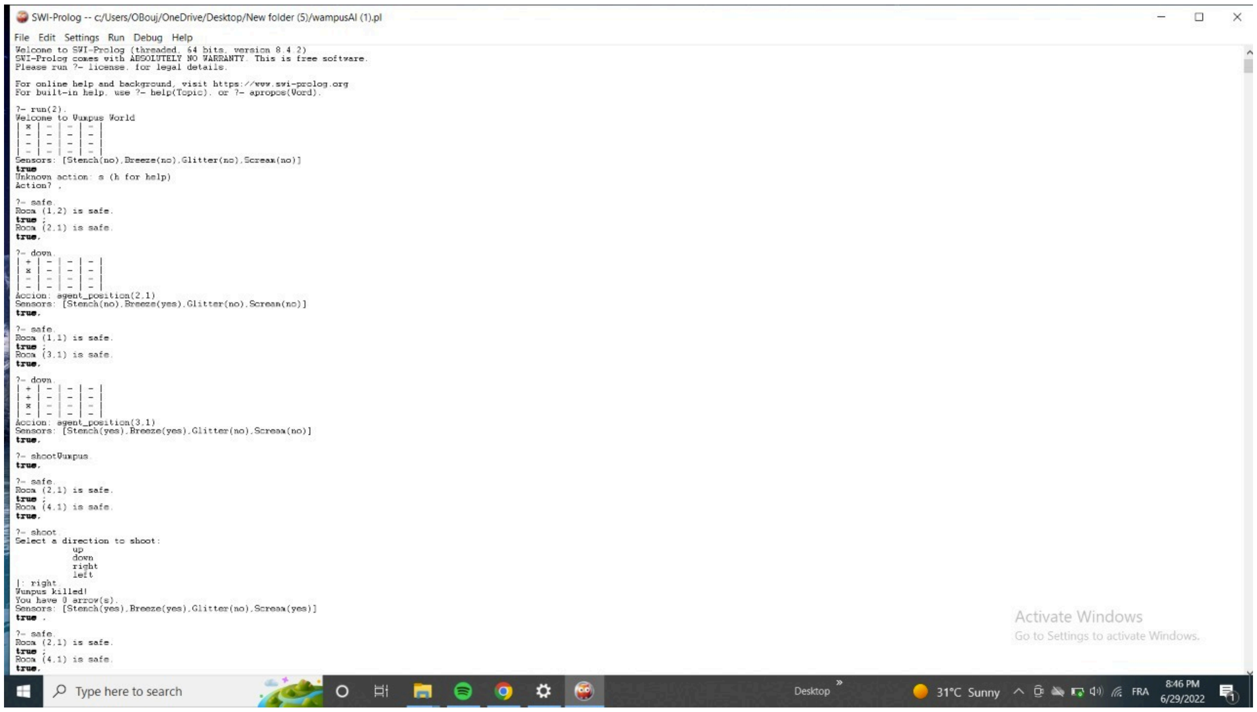
*N.B: **Performance** would be represented in a percentage (%), which can be 33%, 66% or 100%, meaning that if all column (from Alive, Gold picked, Wumpus killed) are true, performance will equal to 100%. If one is false and 2 of them are true, then it is 66%, etc…*

| Worlds | Alive | Gold picked | Wumpus Killed | Performance | Comments |
|---|---|---|---|---|---|
| **N = 1** | True | True | True | 100% | - |
| **N = 2** | True | False | True | 66% | Gold on pit |
| **N = 3** | False | False | False | 0% | Unwinnable |
| **N = 4** | True | True | True | 100% | - |
| **N = 5** | False | False | False | 0% | Unwinnable |
| **N = 6** | True | False | True | 66% | Gold on pit |
| **N = 7** | True | True | True | 100% | Gold on pit |
| **N = 8** | True | False | True | 66% | Gold on pit |
| **N = 9** | True | False | True | 66% | Gold on pit |
| **N = 557** | False | False | False | 0% | Unwinnable |
| **N = 566** | True | True | True | 100% | - |
| **N = 883** | False | True | False | 33% | Unwinnable |
| **N = 50** | True | True | True | 100% | - |
| **N = 54** | True | True | True | 100% | - |
| **N = 87** | True | True | True | 100% | - |

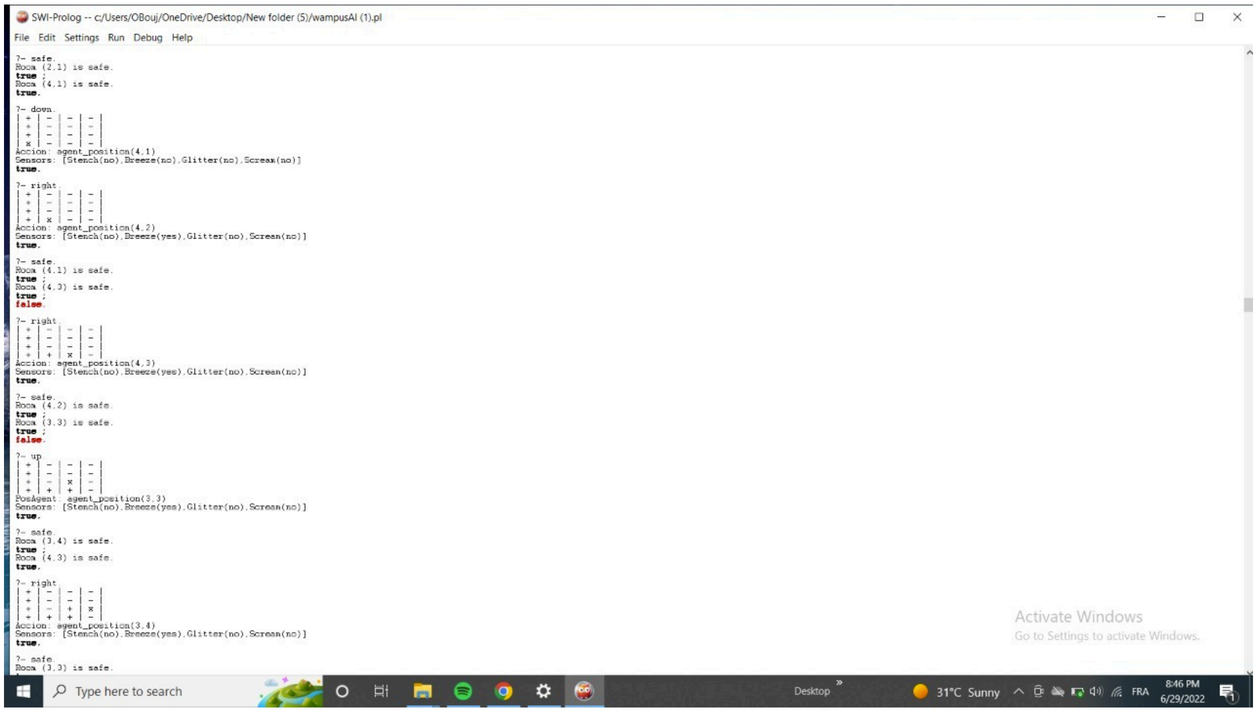
| | | | | | |
|---|---|---|---|---|---|
| **N = 100** | True | False | True | 66% | Gold on pit |
| **N = 105** | True | False | True | 66% | Gold on pit |
| **N = 111** | True | True | False | 66% | Unwinnable |
| **N = 547** | True | False | True | 66% | Gold surrounded by pits |
| **N = 437** | True | True | True | 100% | - |

*<u>Screenshots of some of the experiments:</u>*

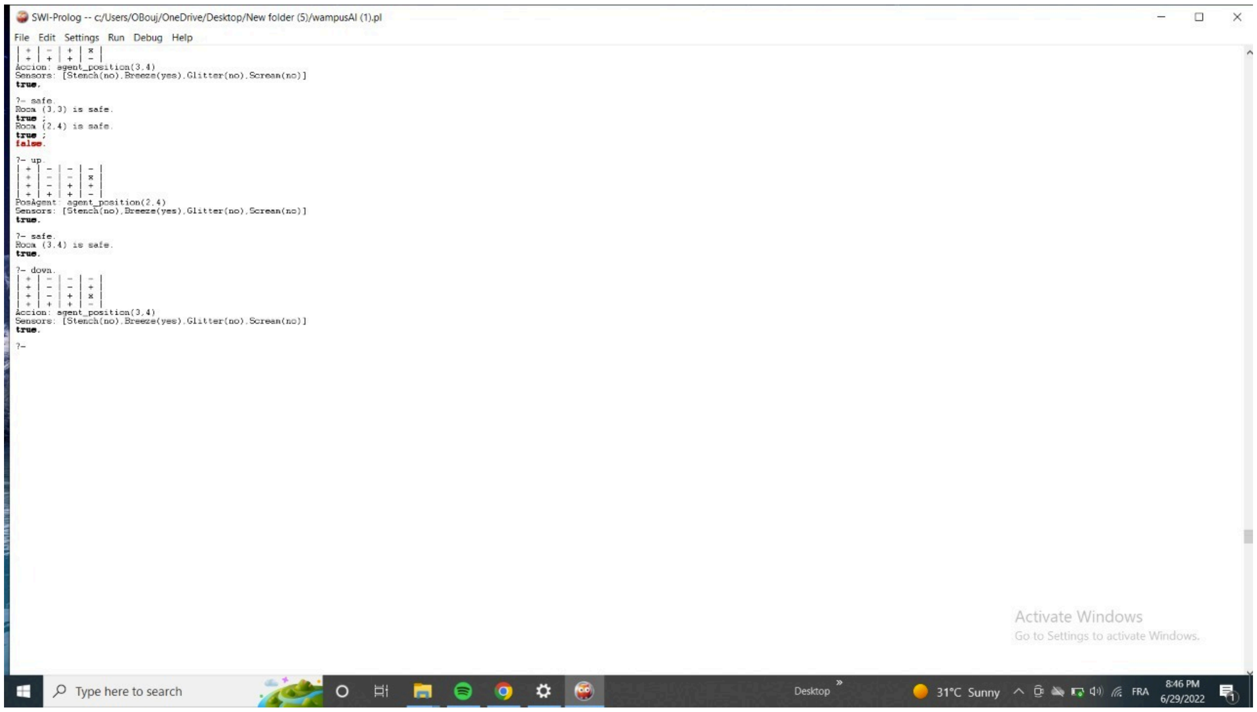
N = 2:


SWI-Prolog -- c:/Users/OBouj/OneDrive/Desktop/New folder (5)/wampusAI (1).pl
File Edit Settings Run Debug Help
Activate Windows
Go to Settings to activate Windows.
Type here to search
Desktop
31°C Sunny
FRA
8:46 PM
6/29/2022

```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- run(2).
Welcome to Wumpus World
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true
Unknown action: s (h for help)
Action? .

?- safe.
Room (1,2) is safe.
true ;
Room (2,1) is safe.
true.

?- down.
| + | - | - | - |
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,1)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (1,1) is safe.
true ;
Room (3,1) is safe.
true.

?- down.
| + | - | - | - |
| + | - | - | - |
| x | - | - | - |
| - | - | - | - |
Accion: agent_position(3,1)
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(no)]
true.

?- shootWumpus.
true.

?- safe.
Room (2,1) is safe.
true ;
Room (4,1) is safe.
true.

?- shoot.
Select a direction to shoot:
            up
            down
            right
            left
|: right.
Wumpus killed!
You have 0 arrow(s).
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(yes)]
true .

?- safe.
Room (2,1) is safe.
true ;
Room (4,1) is safe.
true.
```

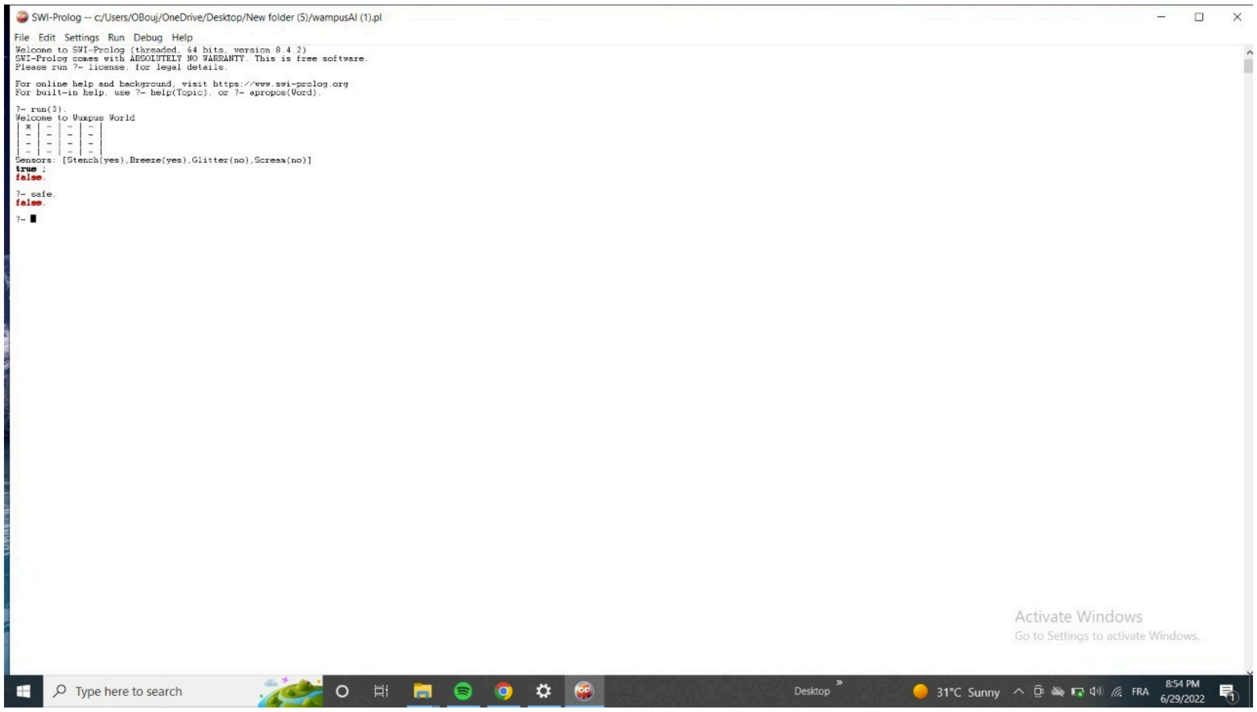

SWI-Prolog -- c:/Users/OBouj/OneDrive/Desktop/New folder (5)/wampusAI (1).pl
File Edit Settings Run Debug Help
Activate Windows
Go to Settings to activate Windows.
Type here to search
Desktop
31°C Sunny
FRA
8:46 PM
6/29/2022

```
?- safe.
Room (2,1) is safe.
true ;
Room (4,1) is safe.
true.

?- down.
| + | - | - | - |
| + | - | - | - |
| + | - | - | - |
| x | - | - | - |
Accion: agent_position(4,1)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- right.
| + | - | - | - |
| + | - | - | - |
| + | - | - | - |
| + | x | - | - |
Accion: agent_position(4,2)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (4,1) is safe.
true ;
Room (4,3) is safe.
true ;
false.

?- right.
| + | - | - | - |
| + | - | - | - |
| + | - | - | - |
| + | + | x | - |
Accion: agent_position(4,3)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (4,2) is safe.
true ;
Room (3,3) is safe.
true ;
false.

?- up.
| + | - | - | - |
| + | - | - | - |
| + | - | x | - |
| + | + | + | - |
PosAgent: agent_position(3,3)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (3,4) is safe.
true ;
Room (4,3) is safe.
true.

?- right.
| + | - | - | - |
| + | - | - | - |
| + | - | + | x |
| + | + | + | - |
Accion: agent_position(3,4)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (3,3) is safe.
```


SWI-Prolog -- c:/Users/OBouj/OneDrive/Desktop/New folder (5)/wampusAI (1).pl
File Edit Settings Run Debug Help
Activate Windows
Go to Settings to activate Windows.
Type here to search
Desktop
31°C Sunny
FRA
8:46 PM
6/29/2022

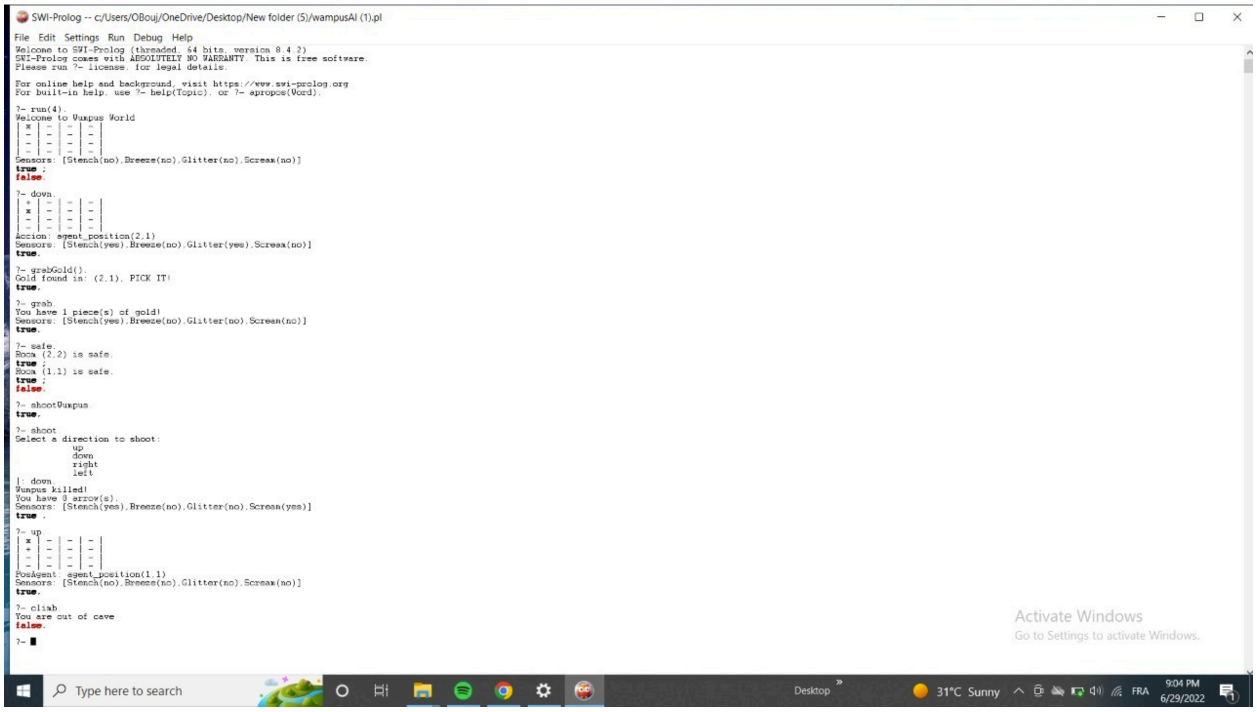
```
| + | - | + | x |
| + | + | + | - |
Accion: agent_position(3,4)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (3,3) is safe.
true ;
Room (2,4) is safe.
true ;
false.

?- up.
| + | - | - | - |
| + | - | - | x |
| + | - | + | + |
| + | + | + | - |
PosAgent: agent_position(2,4)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (3,4) is safe.
true.

?- down.
| + | - | - | - |
| + | - | - | + |
| + | - | + | x |
| + | + | + | - |
Accion: agent_position(3,4)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?-
```

N = 3:



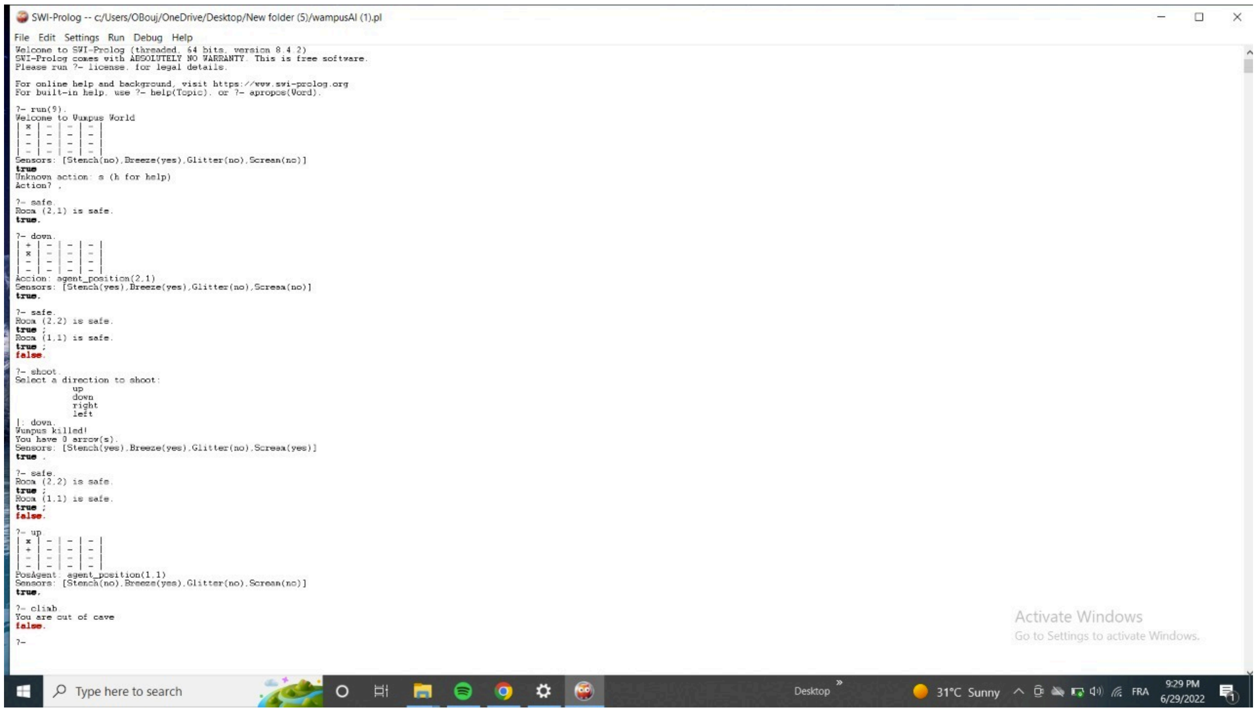
```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- run(3).
Welcome to Wumpus World
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(no)]
true ;
false.

?- safe.
false.

?- 
```

N = 4:



```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- run(4).
Welcome to Wumpus World
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true ;
false.

?- down.
| + | - | - | - |
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,1)
Sensors: [Stench(yes),Breeze(no),Glitter(yes),Scream(no)]
true.

?- grabGold().
Gold found in: (2,1). PICK IT!
true.

?- grab.
You have 1 piece(s) of gold!
Sensors: [Stench(yes),Breeze(no),Glitter(no),Scream(no)]
true.

?- safe.
Room (2,2) is safe.
true ;
Room (1,1) is safe.
true ;
false.

?- shootWumpus.
true.

?- shoot.
Select a direction to shoot:
           up
           down
           right
           left
|: down.
Wumpus killed!
You have 0 arrow(s).
Sensors: [Stench(yes),Breeze(no),Glitter(no),Scream(yes)]
true .

?- up.
| x | - | - | - |
| + | - | - | - |
| - | - | - | - |
| - | - | - | - |
PosAgent: agent_position(1,1)
Sensors: [Stench(no),Breeze(no),Glitter(no),Scream(no)]
true.

?- climb.
You are out of cave
false.

?- 
```

N = 6:



```
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- run(9).
Welcome to Wumpus World
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
| - | - | - | - |
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true
Unknown action: s (h for help)
Action? .

?- safe.
Room (2,1) is safe.
true.

?- down.
| + | - | - | - |
| x | - | - | - |
| - | - | - | - |
| - | - | - | - |
Accion: agent_position(2,1)
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(no)]
true.

?- safe.
Room (2,2) is safe.
true ;
Room (1,1) is safe.
true ;
false.

?- shoot.
Select a direction to shoot:
           up
           down
           right
           left
|: down.
Wumpus killed!
You have 0 arrow(s).
Sensors: [Stench(yes),Breeze(yes),Glitter(no),Scream(yes)]
true .

?- safe.
Room (2,2) is safe.
true ;
Room (1,1) is safe.
true ;
false.

?- up.
| x | - | - | - |
| + | - | - | - |
| - | - | - | - |
| - | - | - | - |
PosAgent: agent_position(1,1)
Sensors: [Stench(no),Breeze(yes),Glitter(no),Scream(no)]
true.

?- climb.
You are out of cave
false.

?-
```

*IV Limitations of the agent:*

In this part, we are going to discuss some of the limitations and issues we found while running the simulations. First, since the agent answers (TELL) only after the has been asked (ASKED), the success or failure will depend heavily on the user being rational. In our case, we estimate the user is rational and will always select a safe room.

Next, we could mention the performance:

- Almost all the experiments gave a performance of 66%, because the agent is being able to kill the Wumpus, but not grab the gold since it is located on a pit.
- If the gold is not on a pit and not surrounded by pits, the agent will always be able to detect it and grab it.
- In case N = 547, the gold was surrounded by pits. The agent won the game without taking the gold.

- Same goes for the Wumpus, for instance in the world N = 111, the Wumpus was surrounded by pits. Thus, the agent could not kill it and win.
- In some cases, the agent can kill the Wumpus and don't grab the goal, where the gold is surrounded by pits.
- In other cases, the agent will be forced to die from without grabbing the gold or killing the Wumpus.

From the experiments above in the table, we can conclude the following:

- **P (Win) = 13/20 = 65%**
- P (Unwinnable) = 5/20 = **25%**
- P (Gold on pit) = 7/20 = **35%**
- P (Gold surrounded by pits) = **5%**
- P (Wumpus surrounded by pits) = **10%**

Thus, the percentage of win rate of the agent is around 65%.

## References:

GitHub - alexroque91/wumpus-world-prolog: Intelligent Agent that plays the Wumpus World using Prolog. (2022). Retrieved 29 June 2022, from https://github.com/alexroque91/wumpus-world-prolog

wumpus-world-prolog/main.pl at master · alexroque91/wumpus-world-prolog. (2022). Retrieved 29 June 2022, from https://github.com/alexroque91/wumpus-world-prolog/blob/master/main.pl

The Wumpus world in Artificial Intelligence - Javatpoint. (2022). Retrieved 29 June 2022, from https://www.javatpoint.com/the-wumpus-world-in-artificial-intelligence

GitHub - krishnangovindraj/wumpusagent: Prolog wumpus world agent for the AI course. (2022). Retrieved 29 June 2022, from https://github.com/krishnangovindraj/wumpusagent

IA-ULL-WumpusWorld/wumpus_game.pl at master · AlbertoCruzLuis/IA-ULL-WumpusWorld. (2022). Retrieved 29 June 2022, from https://github.com/AlbertoCruzLuis/IA-ULL-WumpusWorld/blob/master/wumpus_game.pl