Group Members:
Leila Farah Moussa < 76287 >
Khaoula Ait Soussi < 79155 >

# Wumpus World Project Report

## 1. Introduction:

In this project, we aim at building a knowledge-based agent that can represent a rational player of the Wumpus World game. Our knowledge-based agent is supposed to decide on the next action based on general/accumulated knowledge and current percepts.

The Wumpus World is a 4x4 matrix where each cell represents a room (i.e., 16 rooms). The agent can only move horizontally or vertically to adjacent rooms, and it can only sense the perceptions related to the current room or adjacent rooms.

A room can be empty (i.e., safe), or can have a pit, or can have the Wumpus. If the agent falls in a pit, or moves in a room where the Wumpus is, then the game is over, and the agent loses. According to the instructions of this project, in order to win, the agent has to kill the Wumpus.

The game rules that support the agent to explore the world are as follows:

- The Wumpus causes a stench in the adjacent rooms.
- A pit causes a breeze in the adjacent rooms.
- The gold is in a room where there is glitter.

## 2. Implementation Details:

To implement this project, we followed different approaches to eventually settle for one approach that seemed promising. In both approaches however, we used a set of predicates that will allow the agent to evaluate the safety and risk of the next move.

### 2.1. <u>Explanation of Predicates</u>

⇒ A room **may have a pit** if it is adjacent to a room where there is a breeze. This rule cannot override the next two rules.

⇒ A room **definitely has no pit** if at least one of its adjacent rooms does not have a breeze, or if it was said to have a Wumpus according to another inferencing predicate.

⇒ A room **definitely has a pit** if it is adjacent to a breeze and all the adjacent rooms to that breeze have already been visited or were said to have no pit.

⇒ A room **may have a Wumpus** if it is adjacent to a room where there is a stench. This rule cannot override the next two rules.

⇒ A room **definitely has no Wumpus** if at least one of its adjacent rooms does not have a stench, or if it was said to have a pit according to another inferencing predicate.

⇒ A room **definitely has a Wumpus** if it is adjacent to a stench and all the adjacent rooms to that stench have already been visited or were said to have no Wumpus.

## 2.2. Approach 1: Risk Free (Explorable rooms are the safe ones)

In this approach, we defined an explorable room as one that has no Wumpus and no pit based on the accumulated knowledge.

The list of explorable rooms is initialized to rooms (1,2) and (2,1) since these are adjacent to the start room which has no perceptions, and they are hence safe. Unless the score is zero or the number of iterations reaches 200, the agent repeatedly performs the following operations:

1- Moves from the current position to an explorable room (i.e., 100% safe room)
2- Perceives
3- Tells the knowledge base about the perceptions
4- Asserts to the knowledge base other explorable rooms if any

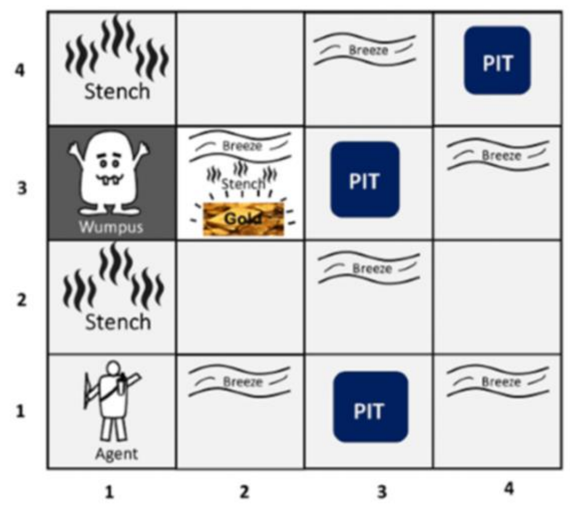
While that is being done, the following operations also take place:

- Whenever an explorable room is visited, it is retracted from the knowledge base.
- Whenever a room is visited, it is asserted to the knowledge base as a visited room.
- The gold is grabbed when the agent perceives a glitter.
- The agent shoots when an adjacent room is said to definitely have a Wumpus.

This approach tries to assure that the agent either wins or gets stuck but never allows the agent to take a risky move. Under this approach these assumptions are made:

1- The list of explorable rooms will most often (i.e., in most configurations) be full, and thus the agent will have a safe room to move to and will rarely get stuck.
2- The explorable rooms will give us enough information to infer the location of the Wumpus, so we do not shoot until we are sure the Wumpus is there.

Unfortunately, we did not fully implement this approach since we decided to switch to the other approach due to lack of time, that is why we do not have output snippets of code execution. However, we tested it on World_A, and although it had slight glitches, it led to killing the Wumpus through the following steps.



Given this world configuration, this approach works and leads the agent to kill the Wumpus.

1- Move to the explorable room (1,2).
2- Retract room(1,2) from the list of explorable rooms and add it to the list of visited rooms
3- Perceive: Agent perceives a stench only.
4- Using the has_pit(_, no) predicate mentioned above, the agent infers that there is no pit in (2,2), and no pit in (1,3) because they are adjacent to a room with no breeze which is the current position. These facts are asserted to the knowledge base, however, since these rooms may have a Wumpus according to has_wumpus(_,maybe) predicate, nothing can be added to the list of explorable rooms.
5- Travel to room(2,1).
6- Perceive: Agent perceives a breeze only.
7- Using has_wumpus(_,no) predicate, the agent infers that there is no Wumpus in (2,2), therefore this room is added to the explorable rooms since it has no Wumpus and no pit.
8- Move to room(2,2)
9- Since no perceptions are sensed, all the adjacent rooms except the previously visited are added to the list of explorable rooms (i.e., room(2,3), room(3,2))
10- Move to room(2,3).
11- Perceive: Agent perceives a stench, a breeze, and a glitter.
12- Grab gold.
13- Shoot room(1,3): Agent shoots room(1,3) because according to has_wumpus(_, yes), this room surely has the Wumpus since it is adjacent to a stench (room(1,2)) and all other adjacent rooms to that stench have already been visited.

## 2.3. Approach 2: Risk Tolerant (Explorable Rooms are assigned a risk cost)

We decided to implement a more risk tolerant approach since the previous method would always fail when there are no safe rooms to explore and there is a need for a cost evaluation.

In this approach we defined explorable rooms as the neighbors of the current room and all previously visited rooms as well as their adjacent rooms (with no duplicates).

To decide which room to move to at each point in the game, the agent chooses based on an assigned cost the represents the level of risk of taking that move (i.e., the least cost the better). Costs are assigned as follows:

- Each room has partial scores and a total score which is the sum of the partial scores
- If a room definitely has a pit or a Wumpus it is assigned the highest score
- If a room may have a Wumpus, it is assigned a partial score of 900.
- If a room may have a pit, it is assigned a partial score of 500.

  Note that a room may be assigned both facts has_wumpus(_, maybe), and has_pit(_,maybe), that is why we have partial scores.
- A safe room is assigned the cost 0.

Unlike the previous approach, the agent in this approach acts based on the outcome of a heuristic function. In the previous approach, the main action was moving to an explorable room, the shoot only took place with certainty. On the other hand, in this approach, the action to be performed is decided upon by a heuristic that takes the perceptions, compares the risk costs of explorable rooms, and decides on the action. So, while the previous approach always fails when there are no safe rooms to visit, this approach may win sometimes if the risky chosen move turns out to be safe.

## 3. Results of The Implemented Approach:

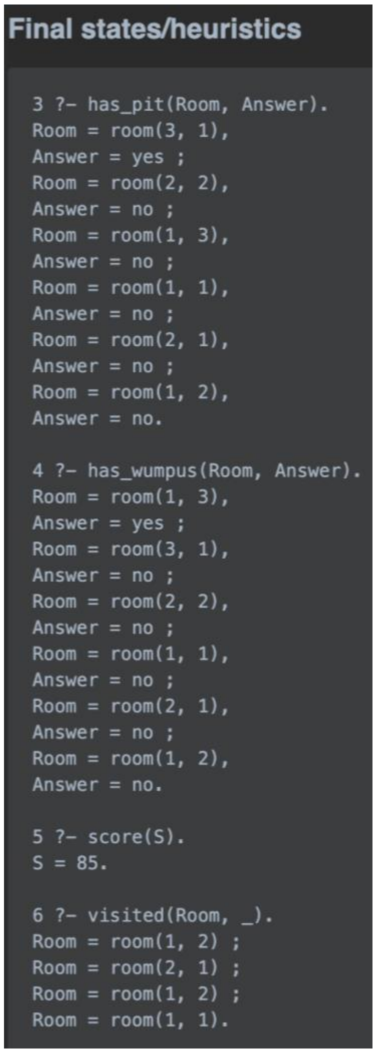
**World_A:**

(Same as the configuration shown in the figure above).

The first configuration we tested is as follows:

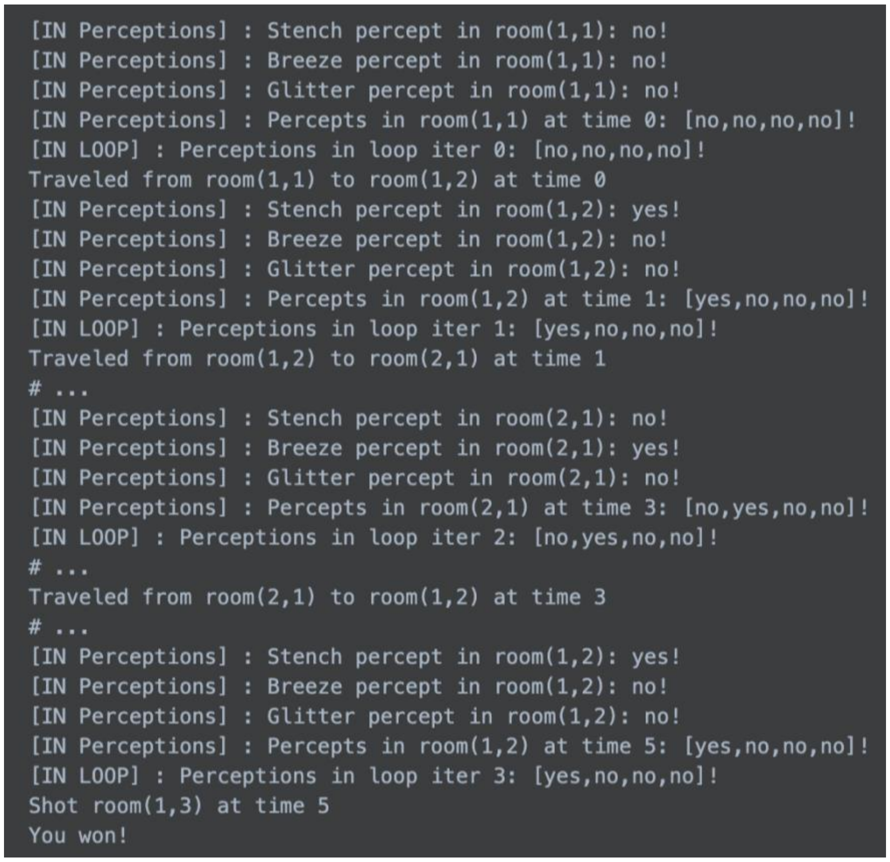
```
wumpus(room(1, 3)).
pit(room(3, 1)).
pit(room(3, 3)).
pit(room(4, 4)).
gold(room(2, 3)).
```

In this configuration and using the second approach, the player succeeded to kill the Wumpus and here is a snippet of the execution results.



Final states/heuristics

```
3 ?- has_pit(Room, Answer).
Room = room(3, 1),
Answer = yes ;
Room = room(2, 2),
Answer = no ;
Room = room(1, 3),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

4 ?- has_wumpus(Room, Answer).
Room = room(1, 3),
Answer = yes ;
Room = room(3, 1),
Answer = no ;
Room = room(2, 2),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

5 ?- score(S).
S = 85.

6 ?- visited(Room, _).
Room = room(1, 2) ;
Room = room(2, 1) ;
Room = room(1, 2) ;
Room = room(1, 1).
```



```
[IN Perceptions] : Stench percept in room(1,1): no!
[IN Perceptions] : Breeze percept in room(1,1): no!
[IN Perceptions] : Glitter percept in room(1,1): no!
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Stench percept in room(1,2): yes!
[IN Perceptions] : Breeze percept in room(1,2): no!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 1: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [yes,no,no,no]!
Traveled from room(1,2) to room(2,1) at time 1
# ...
[IN Perceptions] : Stench percept in room(2,1): no!
[IN Perceptions] : Breeze percept in room(2,1): yes!
[IN Perceptions] : Glitter percept in room(2,1): no!
[IN Perceptions] : Percepts in room(2,1) at time 3: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [no,yes,no,no]!
# ...
Traveled from room(2,1) to room(1,2) at time 3
# ...
[IN Perceptions] : Stench percept in room(1,2): yes!
[IN Perceptions] : Breeze percept in room(1,2): no!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 5: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 3: [yes,no,no,no]!
Shot room(1,3) at time 5
You won!
```

**Analysis and Comparison:**

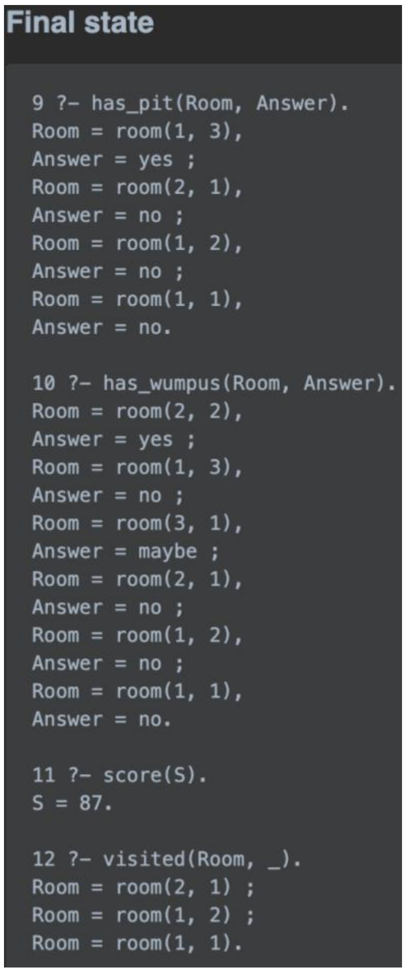
Compared to the previous approach, we can see that this one lead to killing the Wumpus while also visiting the least number of rooms. First because among the explorable rooms it also allows exploring the already visited rooms, that is why the agent could go back and forth between rooms (1,2) and (2,1), second, it could also infer with certainty the place of the Wumpus given the information it gathered.

This configuration shows a case where the agent could easily gather useful information just from the few rooms it visited.

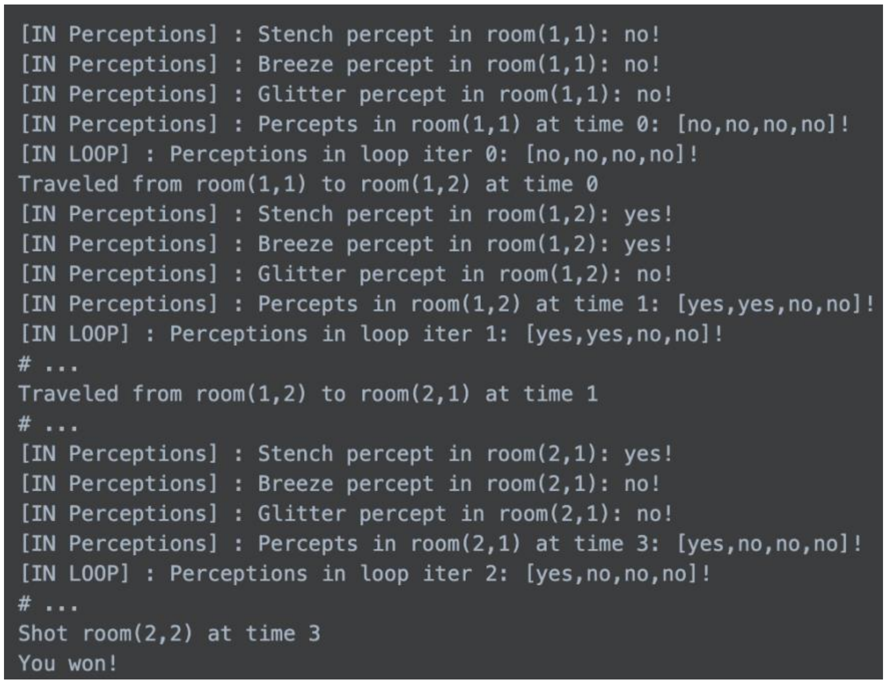
**World_B:**

Similar to configuration A, this configuration allows the agent to have enough information to infer the Wumpus position:

```
wumpus(room(2,2)).
gold(room(3,1)).
pit(room(3,2)).
pit(room(1,3)).
```



```
Final state

9 ?- has_pit(Room, Answer).
Room = room(1, 3),
Answer = yes ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(1, 1),
Answer = no.

10 ?- has_wumpus(Room, Answer).
Room = room(2, 2),
Answer = yes ;
Room = room(1, 3),
Answer = no ;
Room = room(3, 1),
Answer = maybe ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(1, 1),
Answer = no.

11 ?- score(S).
S = 87.

12 ?- visited(Room, _).
Room = room(2, 1) ;
Room = room(1, 2) ;
Room = room(1, 1).
```



```
[IN Perceptions] : Stench percept in room(1,1): no!
[IN Perceptions] : Breeze percept in room(1,1): no!
[IN Perceptions] : Glitter percept in room(1,1): no!
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Stench percept in room(1,2): yes!
[IN Perceptions] : Breeze percept in room(1,2): yes!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 1: [yes,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [yes,yes,no,no]!
# ...
Traveled from room(1,2) to room(2,1) at time 1
# ...
[IN Perceptions] : Stench percept in room(2,1): yes!
[IN Perceptions] : Breeze percept in room(2,1): no!
[IN Perceptions] : Glitter percept in room(2,1): no!
[IN Perceptions] : Percepts in room(2,1) at time 3: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [yes,no,no,no]!
# ...
Shot room(2,2) at time 3
You won!
```

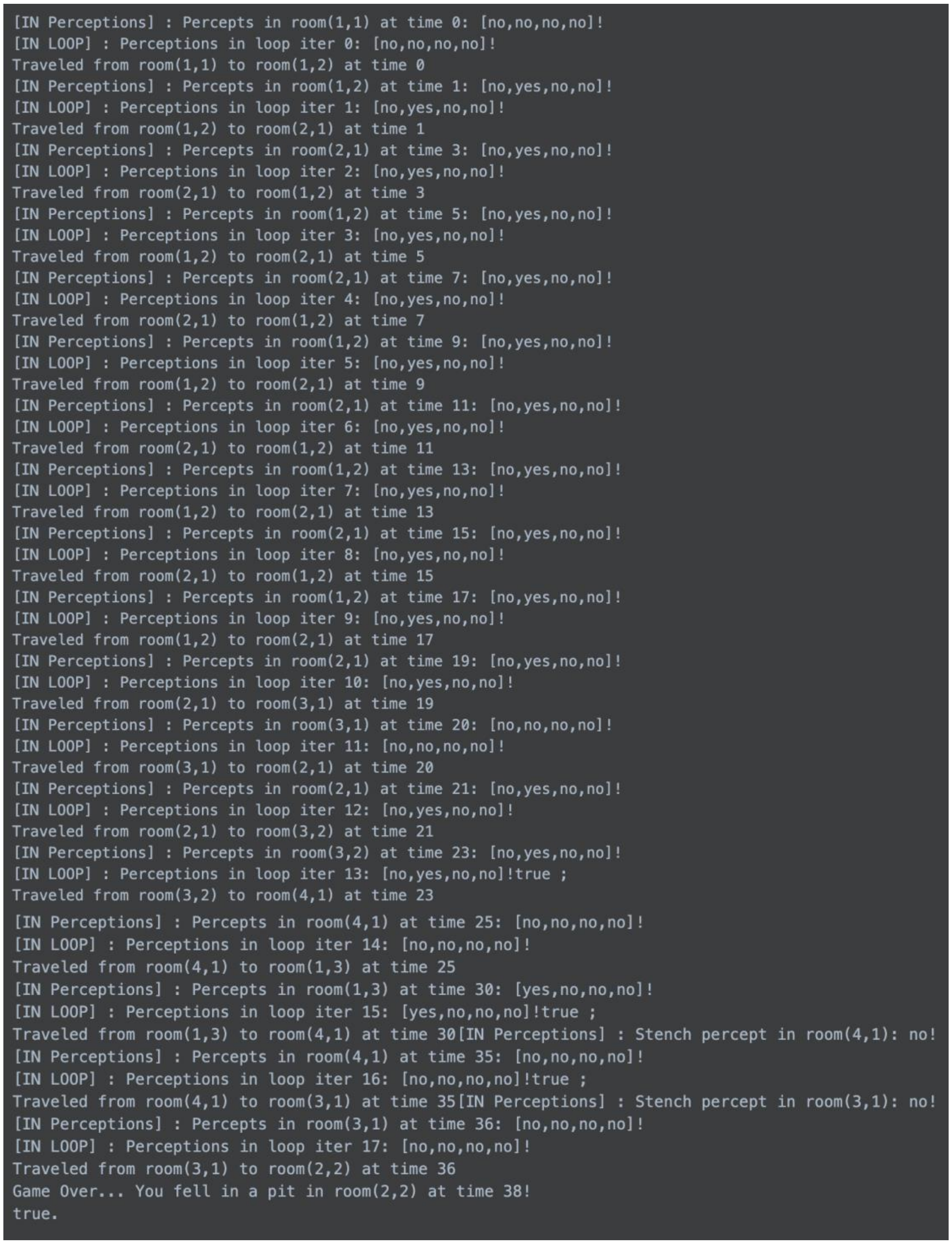
**World_C:**

Unlike the two previous configurations, this third configuration does not allow the agent to certainly deduce where the Wumpus is.

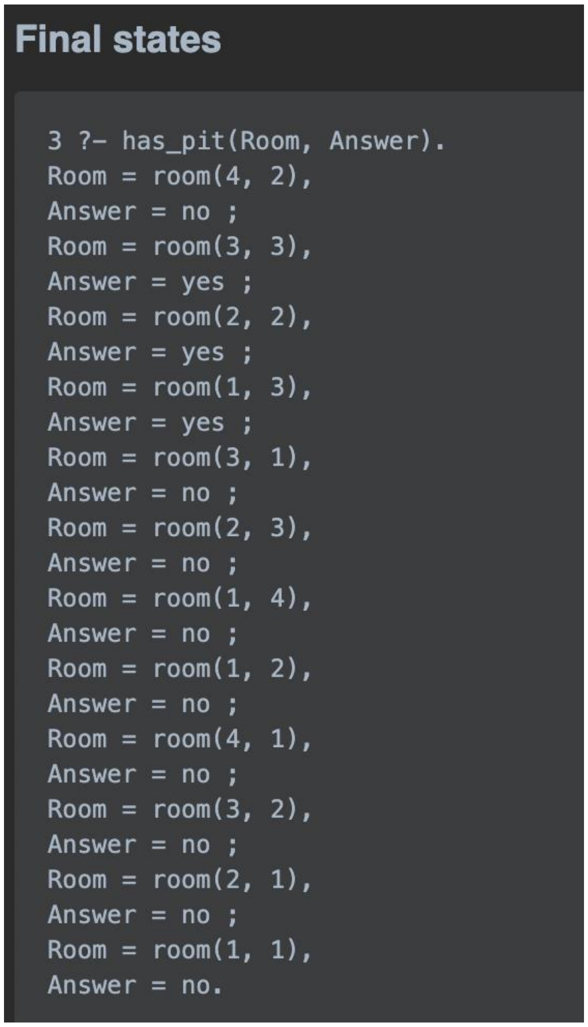
```
wumpus(room(2, 3)).
gold(room(3,3)).
pit(room(2,2)).
pit(room(3,4)).
```

In this configuration, the agent has to take a random risky decision. Depending on the decision taken, the agent may fall in a pit, move to the room of the Wumpus, shoot the wrong room, or win. Here are screenshots of each case:

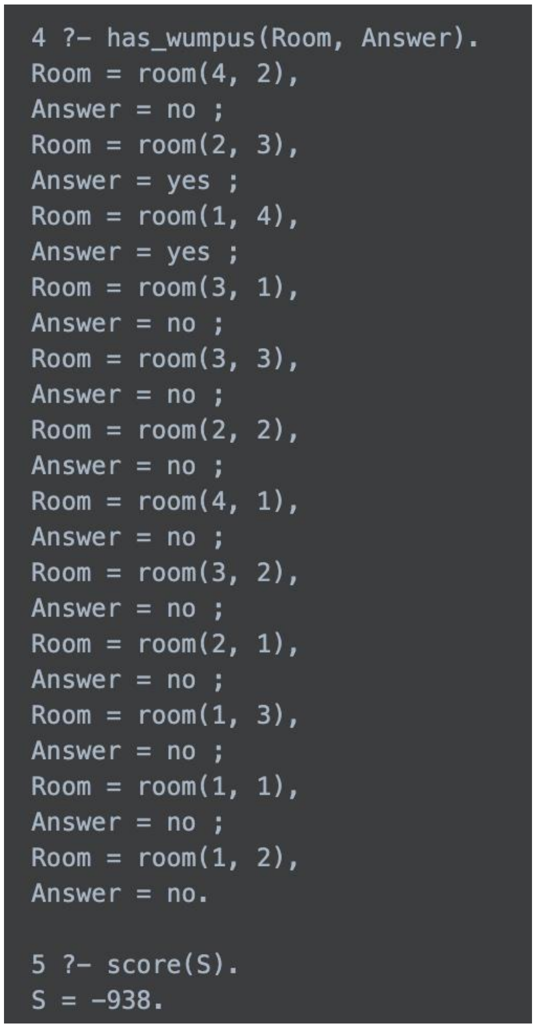
**Agent fall:**



```
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Percepts in room(1,2) at time 1: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 1
[IN Perceptions] : Percepts in room(2,1) at time 3: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 3
[IN Perceptions] : Percepts in room(1,2) at time 5: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 3: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 5
[IN Perceptions] : Percepts in room(2,1) at time 7: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 4: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 7
[IN Perceptions] : Percepts in room(1,2) at time 9: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 5: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 9
[IN Perceptions] : Percepts in room(2,1) at time 11: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 6: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 11
[IN Perceptions] : Percepts in room(1,2) at time 13: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 7: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 13
[IN Perceptions] : Percepts in room(2,1) at time 15: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 8: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 15
[IN Perceptions] : Percepts in room(1,2) at time 17: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 9: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 17
[IN Perceptions] : Percepts in room(2,1) at time 19: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 10: [no,yes,no,no]!
Traveled from room(2,1) to room(3,1) at time 19
[IN Perceptions] : Percepts in room(3,1) at time 20: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 11: [no,no,no,no]!
Traveled from room(3,1) to room(2,1) at time 20
[IN Perceptions] : Percepts in room(2,1) at time 21: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 12: [no,yes,no,no]!
Traveled from room(2,1) to room(3,2) at time 21
[IN Perceptions] : Percepts in room(3,2) at time 23: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 13: [no,yes,no,no]!true ;
Traveled from room(3,2) to room(4,1) at time 23
[IN Perceptions] : Percepts in room(4,1) at time 25: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 14: [no,no,no,no]!
Traveled from room(4,1) to room(1,3) at time 25
[IN Perceptions] : Percepts in room(1,3) at time 30: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 15: [yes,no,no,no]!true ;
Traveled from room(1,3) to room(4,1) at time 30[IN Perceptions] : Stench percept in room(4,1): no!
[IN Perceptions] : Percepts in room(4,1) at time 35: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 16: [no,no,no,no]!true ;
Traveled from room(4,1) to room(3,1) at time 35[IN Perceptions] : Stench percept in room(3,1): no!
[IN Perceptions] : Percepts in room(3,1) at time 36: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 17: [no,no,no,no]!
Traveled from room(3,1) to room(2,2) at time 36
Game Over... You fell in a pit in room(2,2) at time 38!
true.
```

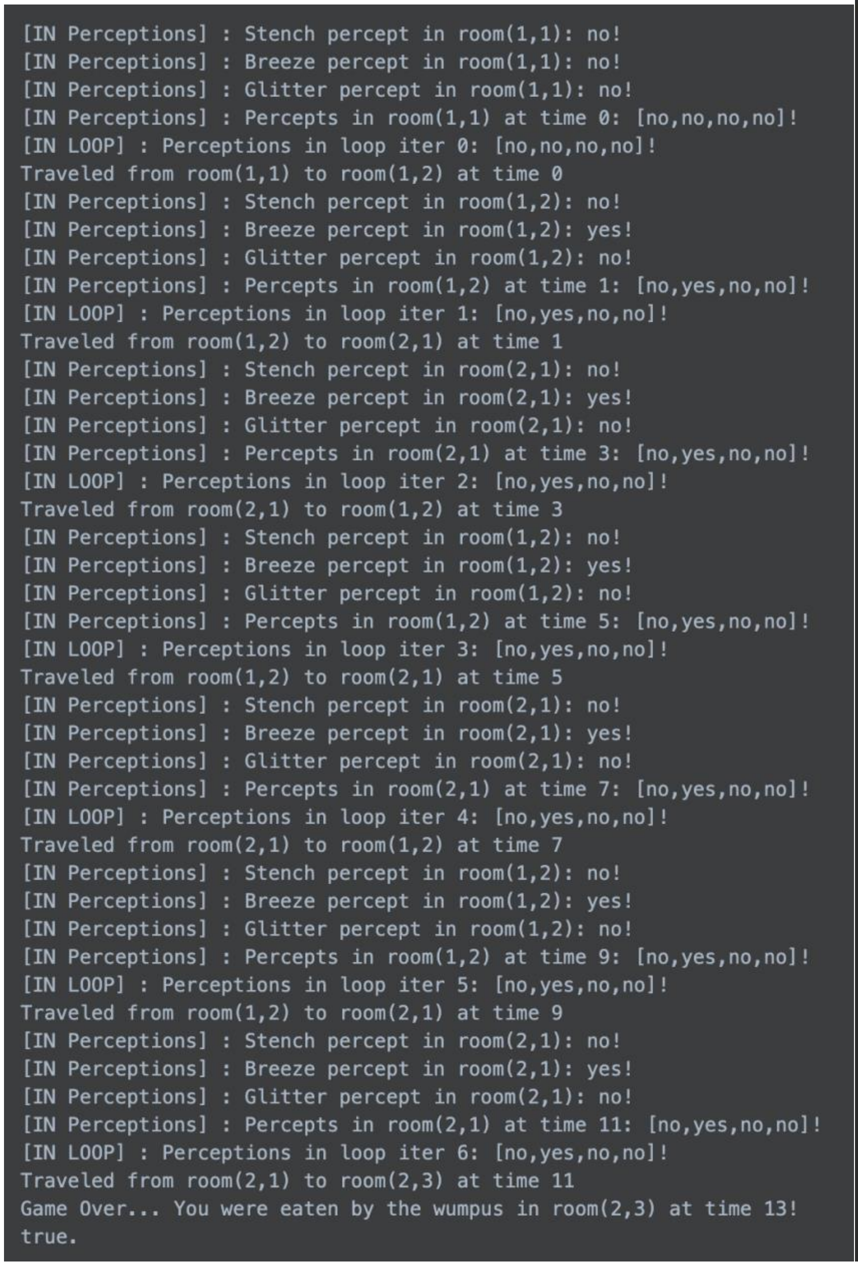

Final states

```
3 ?- has_pit(Room, Answer).
Room = room(4, 2),
Answer = no ;
Room = room(3, 3),
Answer = yes ;
Room = room(2, 2),
Answer = yes ;
Room = room(1, 3),
Answer = yes ;
Room = room(3, 1),
Answer = no ;
Room = room(2, 3),
Answer = no ;
Room = room(1, 4),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(4, 1),
Answer = no ;
Room = room(3, 2),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 1),
Answer = no.
```

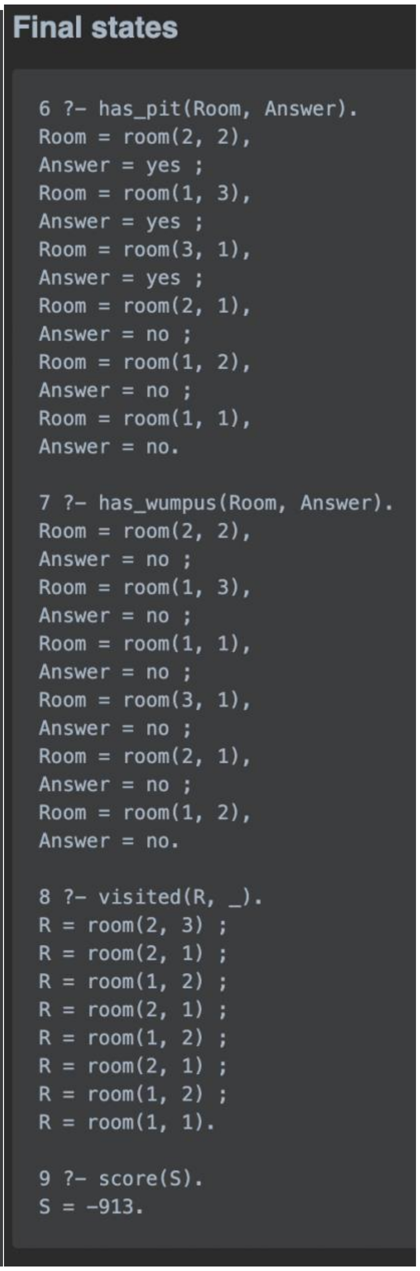


```
4 ?- has_wumpus(Room, Answer).
Room = room(4, 2),
Answer = no ;
Room = room(2, 3),
Answer = yes ;
Room = room(1, 4),
Answer = yes ;
Room = room(3, 1),
Answer = no ;
Room = room(3, 3),
Answer = no ;
Room = room(2, 2),
Answer = no ;
Room = room(4, 1),
Answer = no ;
Room = room(3, 2),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 3),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

5 ?- score(S).
S = -938.
```

**Agent Dead:**



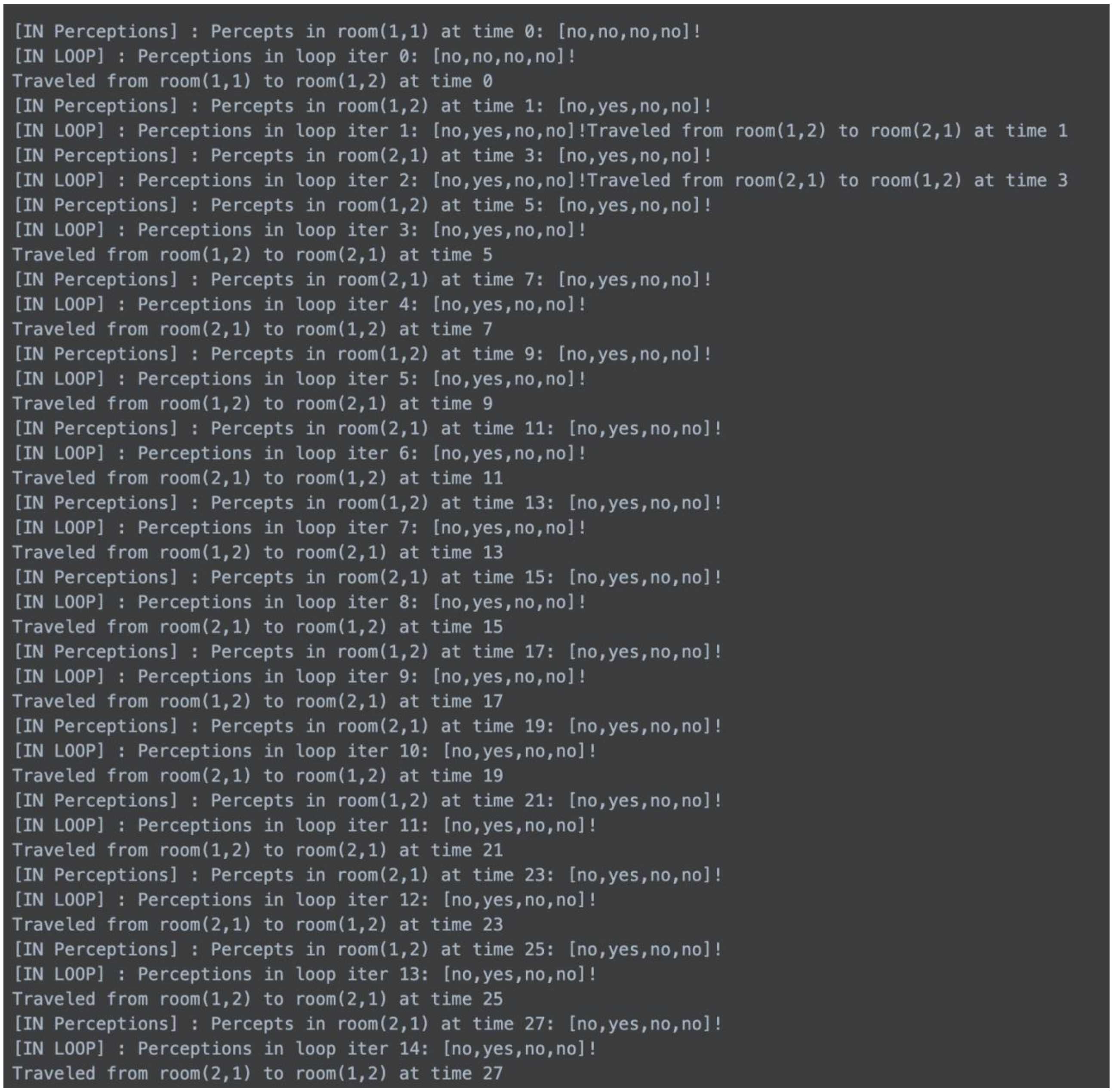
```
[IN Perceptions] : Stench percept in room(1,1): no!
[IN Perceptions] : Breeze percept in room(1,1): no!
[IN Perceptions] : Glitter percept in room(1,1): no!
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Stench percept in room(1,2): no!
[IN Perceptions] : Breeze percept in room(1,2): yes!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 1: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 1
[IN Perceptions] : Stench percept in room(2,1): no!
[IN Perceptions] : Breeze percept in room(2,1): yes!
[IN Perceptions] : Glitter percept in room(2,1): no!
[IN Perceptions] : Percepts in room(2,1) at time 3: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 3
[IN Perceptions] : Stench percept in room(1,2): no!
[IN Perceptions] : Breeze percept in room(1,2): yes!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 5: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 3: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 5
[IN Perceptions] : Stench percept in room(2,1): no!
[IN Perceptions] : Breeze percept in room(2,1): yes!
[IN Perceptions] : Glitter percept in room(2,1): no!
[IN Perceptions] : Percepts in room(2,1) at time 7: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 4: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 7
[IN Perceptions] : Stench percept in room(1,2): no!
[IN Perceptions] : Breeze percept in room(1,2): yes!
[IN Perceptions] : Glitter percept in room(1,2): no!
[IN Perceptions] : Percepts in room(1,2) at time 9: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 5: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 9
[IN Perceptions] : Stench percept in room(2,1): no!
[IN Perceptions] : Breeze percept in room(2,1): yes!
[IN Perceptions] : Glitter percept in room(2,1): no!
[IN Perceptions] : Percepts in room(2,1) at time 11: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 6: [no,yes,no,no]!
Traveled from room(2,1) to room(2,3) at time 11
Game Over... You were eaten by the wumpus in room(2,3) at time 13!
true.
```



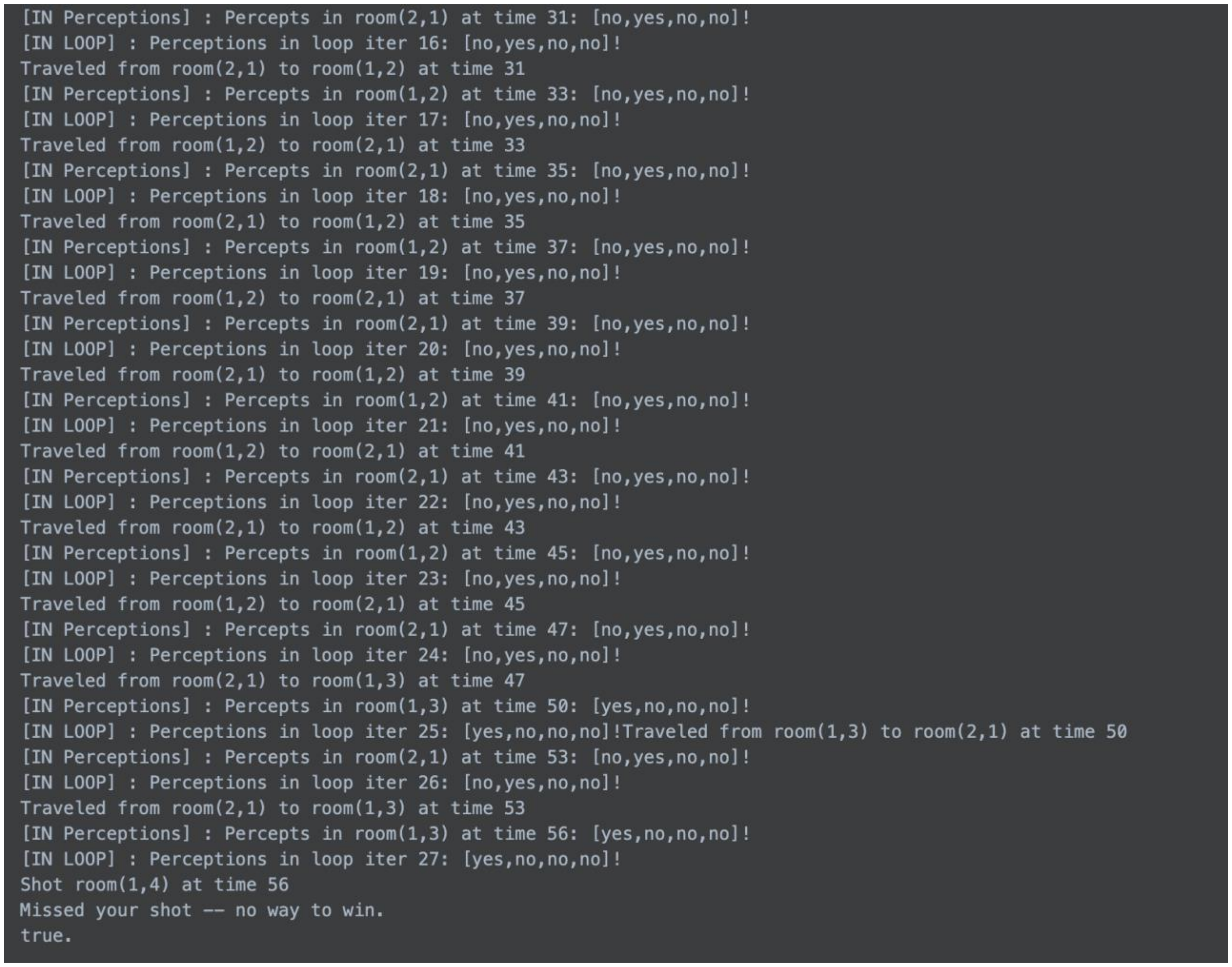
**Final states**

```
6 ?- has_pit(Room, Answer).
Room = room(2, 2),
Answer = yes ;
Room = room(1, 3),
Answer = yes ;
Room = room(3, 1),
Answer = yes ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(1, 1),
Answer = no.

7 ?- has_wumpus(Room, Answer).
Room = room(2, 2),
Answer = no ;
Room = room(1, 3),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(3, 1),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

8 ?- visited(R, _).
R = room(2, 3) ;
R = room(2, 1) ;
R = room(1, 2) ;
R = room(2, 1) ;
R = room(1, 2) ;
R = room(2, 1) ;
R = room(1, 2) ;
R = room(1, 1).

9 ?- score(S).
S = -913.
```

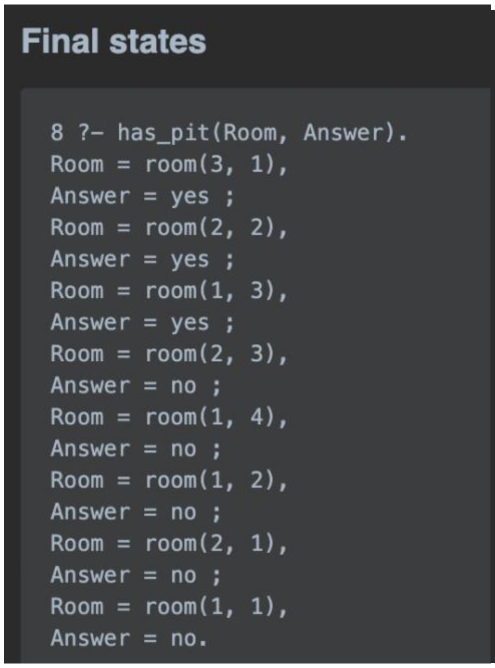
## <u>Wrong Shoot:</u>



```
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Percepts in room(1,2) at time 1: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [no,yes,no,no]!Traveled from room(1,2) to room(2,1) at time 1
[IN Perceptions] : Percepts in room(2,1) at time 3: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [no,yes,no,no]!Traveled from room(2,1) to room(1,2) at time 3
[IN Perceptions] : Percepts in room(1,2) at time 5: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 3: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 5
[IN Perceptions] : Percepts in room(2,1) at time 7: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 4: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 7
[IN Perceptions] : Percepts in room(1,2) at time 9: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 5: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 9
[IN Perceptions] : Percepts in room(2,1) at time 11: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 6: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 11
[IN Perceptions] : Percepts in room(1,2) at time 13: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 7: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 13
[IN Perceptions] : Percepts in room(2,1) at time 15: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 8: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 15
[IN Perceptions] : Percepts in room(1,2) at time 17: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 9: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 17
[IN Perceptions] : Percepts in room(2,1) at time 19: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 10: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 19
[IN Perceptions] : Percepts in room(1,2) at time 21: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 11: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 21
[IN Perceptions] : Percepts in room(2,1) at time 23: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 12: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 23
[IN Perceptions] : Percepts in room(1,2) at time 25: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 13: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 25
[IN Perceptions] : Percepts in room(2,1) at time 27: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 14: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 27
```

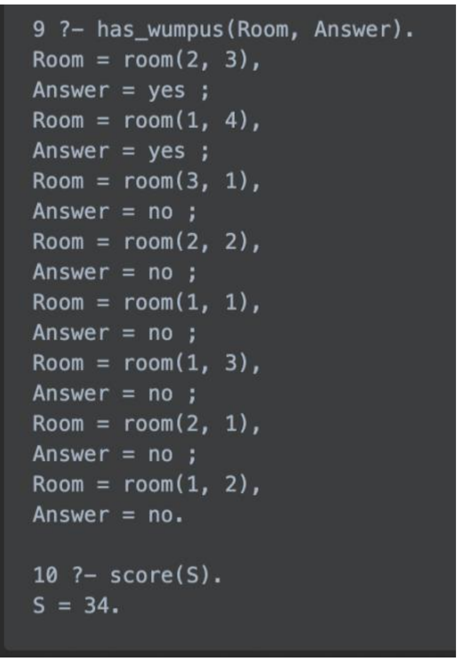


```
[IN Perceptions] : Percepts in room(2,1) at time 31: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 16: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 31
[IN Perceptions] : Percepts in room(1,2) at time 33: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 17: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 33
[IN Perceptions] : Percepts in room(2,1) at time 35: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 18: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 35
[IN Perceptions] : Percepts in room(1,2) at time 37: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 19: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 37
[IN Perceptions] : Percepts in room(2,1) at time 39: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 20: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 39
[IN Perceptions] : Percepts in room(1,2) at time 41: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 21: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 41
[IN Perceptions] : Percepts in room(2,1) at time 43: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 22: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 43
[IN Perceptions] : Percepts in room(1,2) at time 45: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 23: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 45
[IN Perceptions] : Percepts in room(2,1) at time 47: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 24: [no,yes,no,no]!
Traveled from room(2,1) to room(1,3) at time 47
[IN Perceptions] : Percepts in room(1,3) at time 50: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 25: [yes,no,no,no]!Traveled from room(1,3) to room(2,1) at time 50
[IN Perceptions] : Percepts in room(2,1) at time 53: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 26: [no,yes,no,no]!
Traveled from room(2,1) to room(1,3) at time 53
[IN Perceptions] : Percepts in room(1,3) at time 56: [yes,no,no,no]!
[IN LOOP] : Perceptions in loop iter 27: [yes,no,no,no]!
Shot room(1,4) at time 56
Missed your shot -- no way to win.
true.
```



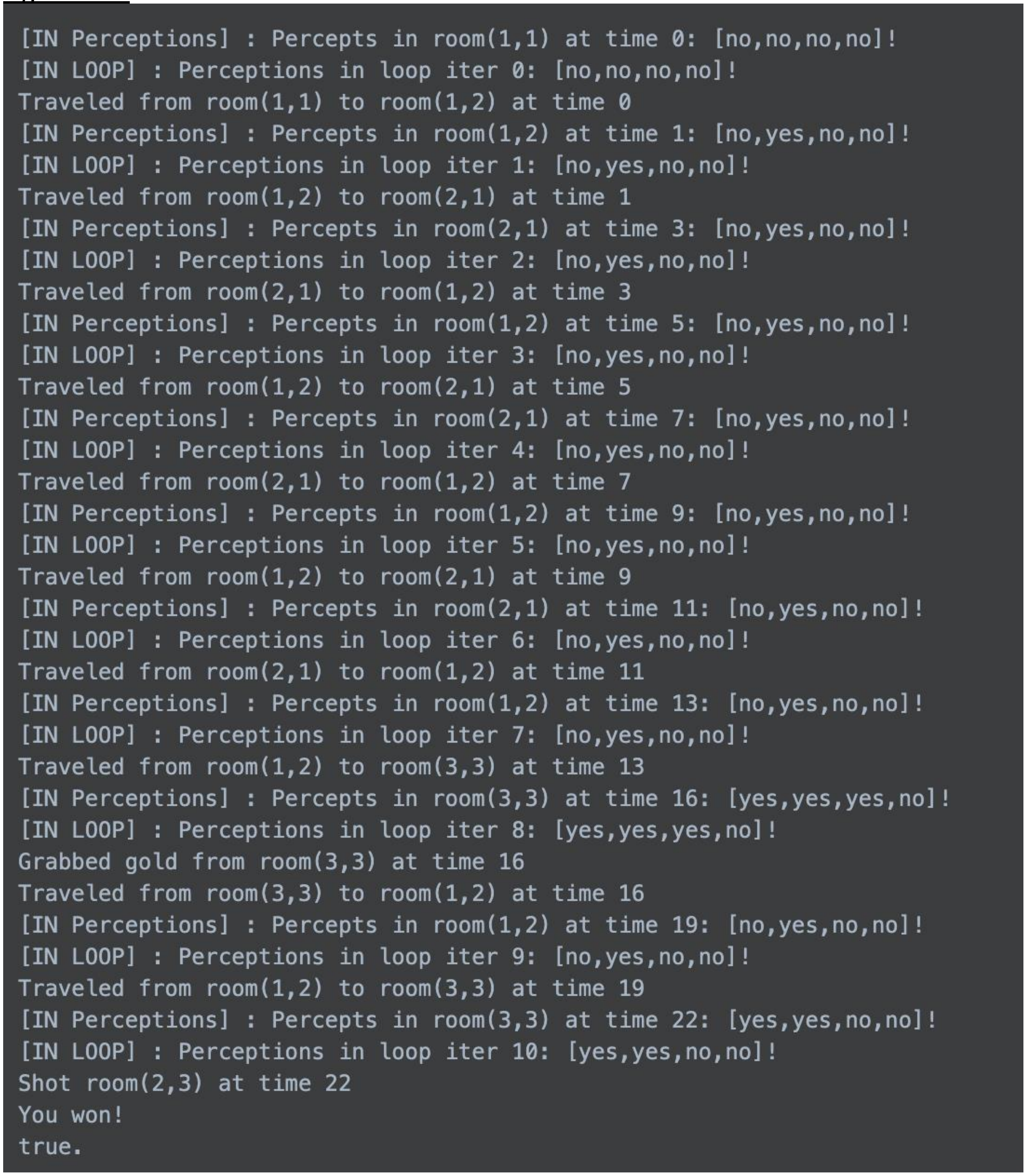
## Final states

```
8 ?- has_pit(Room, Answer).
Room = room(3, 1),
Answer = yes ;
Room = room(2, 2),
Answer = yes ;
Room = room(1, 3),
Answer = yes ;
Room = room(2, 3),
Answer = no ;
Room = room(1, 4),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 1),
Answer = no.
```

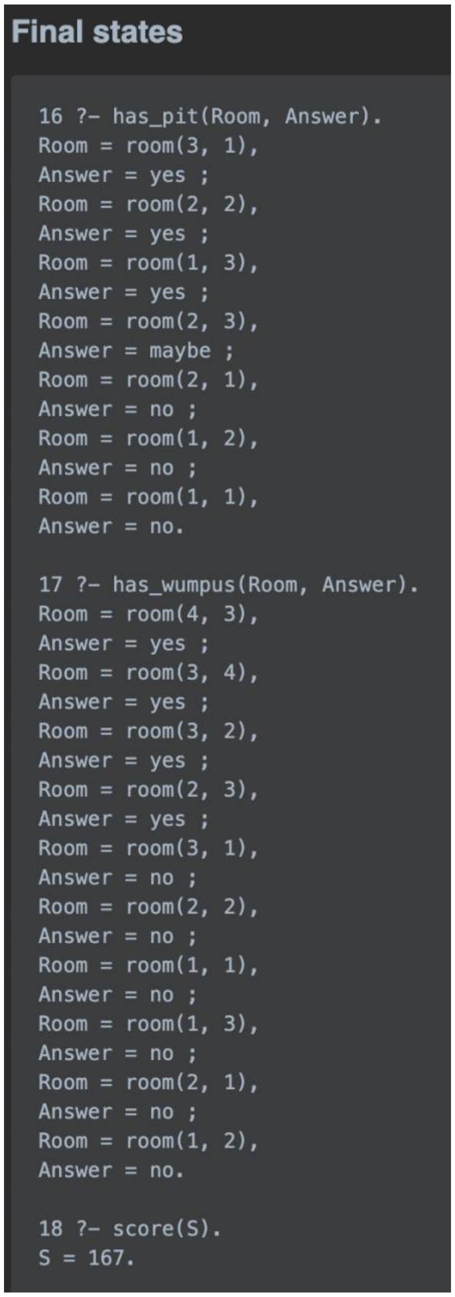


```
9 ?- has_wumpus(Room, Answer).
Room = room(2, 3),
Answer = yes ;
Room = room(1, 4),
Answer = yes ;
Room = room(3, 1),
Answer = no ;
Room = room(2, 2),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(1, 3),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

10 ?- score(S).
S = 34.
```

**Agent Win:**



```
[IN Perceptions] : Percepts in room(1,1) at time 0: [no,no,no,no]!
[IN LOOP] : Perceptions in loop iter 0: [no,no,no,no]!
Traveled from room(1,1) to room(1,2) at time 0
[IN Perceptions] : Percepts in room(1,2) at time 1: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 1: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 1
[IN Perceptions] : Percepts in room(2,1) at time 3: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 2: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 3
[IN Perceptions] : Percepts in room(1,2) at time 5: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 3: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 5
[IN Perceptions] : Percepts in room(2,1) at time 7: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 4: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 7
[IN Perceptions] : Percepts in room(1,2) at time 9: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 5: [no,yes,no,no]!
Traveled from room(1,2) to room(2,1) at time 9
[IN Perceptions] : Percepts in room(2,1) at time 11: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 6: [no,yes,no,no]!
Traveled from room(2,1) to room(1,2) at time 11
[IN Perceptions] : Percepts in room(1,2) at time 13: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 7: [no,yes,no,no]!
Traveled from room(1,2) to room(3,3) at time 13
[IN Perceptions] : Percepts in room(3,3) at time 16: [yes,yes,yes,no]!
[IN LOOP] : Perceptions in loop iter 8: [yes,yes,yes,no]!
Grabbed gold from room(3,3) at time 16
Traveled from room(3,3) to room(1,2) at time 16
[IN Perceptions] : Percepts in room(1,2) at time 19: [no,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 9: [no,yes,no,no]!
Traveled from room(1,2) to room(3,3) at time 19
[IN Perceptions] : Percepts in room(3,3) at time 22: [yes,yes,no,no]!
[IN LOOP] : Perceptions in loop iter 10: [yes,yes,no,no]!
Shot room(2,3) at time 22
You won!
true.
```



Final states

```
16 ?- has_pit(Room, Answer).
Room = room(3, 1),
Answer = yes ;
Room = room(2, 2),
Answer = yes ;
Room = room(1, 3),
Answer = yes ;
Room = room(2, 3),
Answer = maybe ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no ;
Room = room(1, 1),
Answer = no.

17 ?- has_wumpus(Room, Answer).
Room = room(4, 3),
Answer = yes ;
Room = room(3, 4),
Answer = yes ;
Room = room(3, 2),
Answer = yes ;
Room = room(2, 3),
Answer = yes ;
Room = room(3, 1),
Answer = no ;
Room = room(2, 2),
Answer = no ;
Room = room(1, 1),
Answer = no ;
Room = room(1, 3),
Answer = no ;
Room = room(2, 1),
Answer = no ;
Room = room(1, 2),
Answer = no.

18 ?- score(S).
S = 167.
```

**Future Work:**

I personally hoped we could work more on the first approach and get it to support evaluation of neighboring rooms when there are no explorable rooms. I believe the best plan is to extend on the work that we've already done by leveraging both approaches. From what I noticed in other configurations I ran; the first approach prioritizes safety while the second one prioritizes risk tolerance. We can build a solution that does prioritize safety over risk but still takes risk when necessary.