German University in Cairo
Department of Computer Science
Dr. Haythem O. Ismail

Introduction to Artificial Intelligence, Winter Term 2016
Project 2: $\forall \mathbf{x} \exists \mathbf{a} \exists \mathbf{s}[\mathbf{Catch}(\mathbf{I}, \mathbf{x}, \mathbf{Result}(\mathbf{a}, \mathbf{s}))]$

Due: November 21^{st} *at 23:59*

1. Project Description

In this project, you will design and implement a logic-based version of the agent you implemented in Project 1. A logic-based agent operates by storing sentences about the world in its knowledge base, using an inference mechanism to infer new sentences, and using these sentences to decide which action to take. You may use any **logic** programming language (choices include but are not limited to Prolog, SNePSLOG, GOLOG, and CHR).

To implement this agent correctly, you should follow the following steps.

a) Modify the `GenMaze` function you implemented in Project 1 to construct logical sentences of the situation calculus representing (i) the configuration of the maze, (ii) the starting and ending points of the agent, (iii) the locations of the pokémons, and (iv) the time units needed to hatch the egg. All of the constructed logical sentences should be written to an external file. This file represents part of the agent's knowledge base and should be loaded in the agent's program.

b) For each fluent, write a successor-state axiom and add the axioms to the agent's KB.

c) Query the agent's KB to generate a plan that the agent can follow to go from the start to the end point, collect all the pokémons, and hatch the egg.[1] The result of the query should be a situation described as the result of doing some sequence of actions from the initial situation S_0.

2. Groups:

You may work in groups of *at most* three.

3. Deliverables

a) Source Code

- The modified implementation of `GenMaze`, together with *at least two* output files containing logical sentences describing two different maze configurations.
- The agent's program loading the output file from `GenMaze`, and containing the implementation of the successor-state axioms. You should include in this file, as a comment, the query that you wrote to generate the plan.
- Part of the grade will be on how readable your code is. Use explanatory comments whenever possible

[1] Similar to Practice Assignment 5: Exercise 5-1 (d).

b) Project Report, including the following.

- A brief description of your understanding of the problem.
- A description of the modified implementation of `GenMaze`.
- A discussion of the syntax and semantics of the action terms and predicate symbols you employ.
- A discussion of your implementation of the successor-state axioms.
- A description of the query used to generate the plan.
- At least two running examples from your implementation.
- If your program does not run, your report should include a discussion of what you think the problem is and any suggestions you might have for solving it.
- Proper citation of any sources you might have consulted in the course of completing the project. *Under no condition*, you may use on-line code as part of your implementation.

4. Important Dates

Source code. On-line submission by November 21 at 23:59 using the "Submission" link on the course web site.

Project Report. A hard-copy should be submitted. You have two options:

a) November 21 by 16:00;

b) November 22 by 16:00 provided that an *identical* on-line version is submitted with the code (by November 21 at 23:59).

Brainstorming Session. In tutorials.