# Workshop Lecture 2

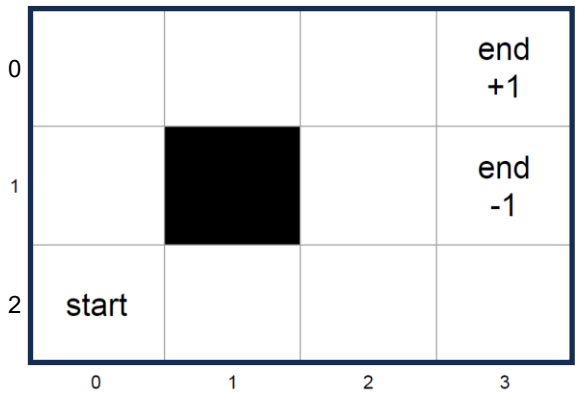
## Part A: A simple maze environment

In the Blackboard folder for this week, you will see the python script "`lecture2-simulation.py`" under the workshop materials. Save locally, and run with (in terminal, `cd` to the location of the script, Python 3 assumed):

```
python3 lecture2-simulation.py
```

This script defines a simple maze (image below), and a simple agent, with the task of learning how to get from the start to the end goal. Take a few minutes to understand the structure of the code (link to base code provided in script). You will notice that there is an 'agent' implemented: `Agent` instantiates a simple value iteration algorithm over states (not including transitions).

In the discrete environment, the starting point is always position (2,0). There are two terminal states: one provides a reward of +1, and the other provides a reward of -1. The black box indicates an impassable state (represented by 'X' on the console representation of the maze), and the edges of the environment are also not passable (i.e. this is not a toroidal environment).




## Part B: Basic Learning Agent

At the bottom of the script, in `main`, you will see an `Agent` instantiated, and 50 iterations (episodes) of completing the maze (this can be changed with the global parameter `L_ITERATIONS`). There are a few tasks to consider regarding this basic learning agent:

1. Familiarise yourself with this agent and in particular the `play()` method. This is the main learning loop in which action selection takes place. In later weeks, we will explore this to integrate human feedback, so it is worth understanding now.

2. Run this simulation and see what results are produced at the end of the simulation (i.e. after the agent has been through the set number of episodes – note that each episode is not independent, but leaves the agent with the information it has learned, but resets its position in the maze). Understand what these results mean in the context of the learning process and agent performance. Vary parameters (leave the environment size/characteristics unchanged) and explore the impact, comparing performance between different parameter settings. Parameters to consider in particular (understand what each one means first):
    - `EXPLORE`: modifying the trade-off between exploration and exploitation
    - `L_ITERATIONS`: number of training episodes the agent undertakes

3. The learning algorithm learns a value for each state, but does not consider the transitions between states. Consider how this compares with standard Q-learning, and the advantages/disadvantages of the approach used in this script (*Hint: it is not a very good algorithm…*). Consider modifying the `Agent` to instantiate Q-learning proper, or a similar Reinforcement Learning algorithm. This can become a valuable benchmark algorithm when it comes to the assignment associated with the HRI part of this module.