[Week 2 - Lab - Welcome to the Grid!](https://www.ole.bris.ac.uk/webapps/blackboard/content/listContent.jsp?course_id=_257207_1&content_id=_8044794_1&mode=reset)

**Intro to GridWorld**

Welcome to the Week 2 Artificial Intelligence Lab! The purpose of this lab is to introduce you to the GridWorld platform. This lab is **vitally** important for both coursework and exam students so please make sure you complete it.

You will need to use a local swipl installation in order to run this code as SWISH **will not work**. You are also encouraged to use a text editor of your choice to edit the files (Both VSCode and Atom have support for prolog). If you have any trouble setting up swipl then please come to a lab or make a post in the forum.

You are provided with library code that runs the local webserver you will use to interact with the GridWorld. This is all stored inside /ailp (standing for Artificial Intelligence with Logic Programming).

You are also provided with 2 files of the form lab\_\*\_1234567.pl where you will need to write your lab solutions by completing the predicate stubs. You may replace the 1234567 in the filenames with *your own* 7-digit student number (this will be required if you do the coursework but, since these labs aren't assessed, you are free to leave it unchanged or any other NUMBER that you want).

You won't need to look into the **internals**of the library but you may do so, if you want. One thing you might want to do is temporarily change the default **grid size** from 10 to a smaller number such as 5 to make it easier to debug your code. This can be done by changing "X=10" to "X=5" in lines 52 and 53 of /ailp/library/game\_predicates.pl; and by changing "X=25" to "X is "N\*N//4" in line 79. This will allow you to make the grid a small as 3\*3 (but you may encounter issues if you try to go lower). Please ensure that you fully **restart**Prolog if you make and changes to the library code.

Please read the Introduction to Gridworld **guidance notes** attached underneath the lecture slides.

**Prolog: Lab Grid**

For this exercise you will learn how to move around the grid world and ultimately use you agent to draw a spiral. The grid world is just the name of the environment that these labs will take part in. For this exercise you will be editing lab\_grid\_1234567.pl.

**Possible Queries**

The following queries have been exported by the library code:

| **Query Name** | **Description** |
| --- | --- |
| start | Initiate the web server |
| shell | Open the command shell |
| my\_agent(-A) | Returns the ID of your Agent |
| ailp\_grid\_size(-N) | Returns the size of the grid |
| get\_agent\_position(+A,-Pos) | Returns the position of Agent A |
| agent\_do\_moves(+A,+L) | Makes Agent A perform the list of moves L |

**How to run?**

**THIS IS ONE OF THE MOST IMPORTANT SECTIONS**. In order to run the grid world, you need to open ailp.pl using swipl. This exercise is known as lab grid so we will pass that as a command line argument (e.g. swipl ailp.pl lab grid).

Upon loading you will likely see a lot of warnings about singleton variables. This is because the skeleton definitions in lab\_grid\_1234567.pl have not yet been properly implemented.

Next, Type in start. and hit enter to start the webserver. If you press anything but 'N'/'n', then the webpage will open automatically in your default browser. **Hit the run button on the webpage before you continue.**

We will use a shell to initialise the grid so type in shell. and then setup. in the following command prompt. (While the main prolog prompt is ?-, in the shell it will look like ?).

After the setup/0 has finished, you can exit the shell using either stop.,CTRL+D or CTRL+C then (a)bort at the prompt.

**Identifying your Agent**

At the top left of the grid, you will be able to see a coloured shape. This is your agent. For the first task in this exercise, you need to find a query that allows you to identify your agent and their current position.

**Create predicate m/1**

The first step of moving around the grid world is defining the directions that we are able to move. They are n,e,s and w. For the first part of this exercise, you will need to define the predicate m(A) such that it is true when A is a valid direction.

**Create predicate on\_board/1**

Next, We need to create the predicate on\_board(+P) that tells us if a position P is on the board. Positions are stored as a compound term p(X,Y) where both X and Y range from 1 to the grid size inclusive.

TIP: You can use the predicate ailp\_grid\_size/1 to find the size of the grid.

**Create predicate pos\_step/3**

For this step we need to find the new position after an agent has made a possible step (although we don't need to check the validity yet!). The full predicate is pos\_step(+Pos,+Dir,-NPos).

**Create predicate new\_pos/3**

We are now able to combine the previous steps together to find the new position after a move has been made as well as to check that the move is valid. new\_pos(+Pos,+M,-NPos) should be true if moving from Pos in direction M will lead to NPos and NPos is on the board.

**Create predicate complete/1**

The predicate complete(+L) should be true if the Length of L is equal to the number of cells in the grid. The size of the grid **SHOULD NOT** be hardcoded.

**Create predicate spiral/1**

The final part of this exercise is to implement the predicate spiral(-L) which will move the agent from the start position p(1,1) towards the center of the grid in a spiral pattern. L should be the list of steps that are taken by the agent to reach the center **including** the start position (You can test this using complete/1)

TIP: In order for the agent to actually move, you will need to use agent\_do\_moves(A,L) where A is the identifier of your Agent and L is a path **that does not include** your current position