# Tutorial 1

## Name: Zhaokun Su

## Breakout group: 3

## Zid: z5235878

**Discussion results for group during tutorial time:**

A screenshot of a cell phone

Description automatically generated

**(i)  This agent is more complex than a reactive agent that merely acts on the current percepts on each cycle. In what environments does this agent have an advantage over the reactive agent? Run some experiments to test your ideas.**

The agent in this exercise is more complicated than a reactive agent. It is more intelligent because this agent is goal-oriented, the agent has a horizon to find some food within a distance rather than just follow the complete condition-action rules to choose its next move. It is goal based. Besides that, it can be shown that this agent can record its behaviors and this is another reason why this agent is more complex.

As we can see, if we change the horizon to 100 (as if the environment is fully-observable), the result score will be better:

A picture containing drawing, meter

Description automatically generated

A close up of a logo

Description automatically generated

And if we change the horizon to 0 ( as if no goals to discover), the result score will be so bad:

A close up of text on a black background

Description automatically generated

Eventually, we can conclude that this agent has a higher level compared with a reactive agent, no matter in which environment, this agent will perform better.

And when the horizon distance increase, foods are more easily to be discovered, so this agent has an advantage over the reactive agent.

**(ii)  This agent stores all previous percepts in a set but does not update that set when food is observed to disappear (food that should be possible to see is no longer visible). Change the code to implement this functionality.**

We need to add self.food.remove() at the end of the else block so that delete coresponding food in our food set.

A screenshot of a cell phone

Description automatically generated

**(iii)  (Harder) Consider how to define the agent action selection function when there are multiple agents in the world, all competing for food. Code the perception and action selection functions to handle this scenario and run some experiments. Note: It is straightforward using the existing code to create any number of agents.**

It mainly logic can be like this:

There are multiple agents participating in this competition, and they both have the same horizon. Since the food is randomly appearing or disappearing, so:

1. When this food has been reached firstly by agent A, score A + 1, and this food disappears in the environment, a new food appears in anther location somewhere in this environment.
2. If two agents or more than two agents reach the same food at the same time. All of those food score + 1 and this food disappears in the environment, a new food appears in anther location somewhere in this environment.
3. It makes sense that different agents can have different horizons.