Fall 2022 - CSC545/645 Artificial Intelligence - Assignment 1 Due date: Tuesday, September 6, 2022, 2pm. Hint: you might want to consult also the AIMA web pages: http://aima.cs.berkeley.edu/ for the online code repository.

Exercise 1.1 (10 points)

Read chapters 1 (Introduction) and 2 (Intelligent Agents) of the textbook.

1. **Define in your own words the following terms: agent, agent function, agent program, rationality, autonomy, reflex agent, model-based agent, goal-based agent, utility-based agent, learning agent.**
   [5 points]

   Agent:
   A thing that has the ability to act upon or make decisions based on its perception of its environment.

   Agent Function:
   An abstract way to consider how an AI should map a given input into an output. Critically this is not the hard-coded logic but instead a way of thinking about the various outputs an AI can have and should enact upon based on various different inputs. It is easy to consider this as a table of inputs for simple programs but this grows almost infinitely as the AIs task gets more complex.

   Agent Program:
   The hard-coded logic that programmers write, based on the agent function map, in an attempt to have a machine make desired outputs based on various inputs.

   Rationality:
   A definition used to describe the intelligence of a machine based on its ability to determine what makes the most ‘sense’ (generally through the use of statistical models) to do at a certain time given its interpretation of an environment to achieve the most optimal outcome.

   Autonomy:
   Having the ability to learn about the environment and predict an outcome due to lack of given knowledge by the designer to the machine.

Reflex Agent:
An agent that acts in the moment based on the most current input only. It does not care about previous events and acts according to what is most optimal at the current time. It is incredibly simple, given certain conditions a certain action can be executed.

Model-Based Agent:
An agent that makes decisions based on keeping an updated model of what it believes the world around it looks like and making reflexive judgments based on that model.

Goal-Based Agent:
Built upon the same ideas of Model-Based Agents by maintaining a mock image of the world, these agents also consider how their environment will change given a certain interaction and attempt to think ahead about which chain of actions will lead to the desired goal. These agents are greatly affected by both the effectiveness of their search algorithm and the description of the goal.

Utility-Based Agent:
Agents of this type still interpret their surroundings into a model and determine sets of actions that would lead them to their goal, but now also determine the effectiveness of these various paths among any parameter deemed important. These agents can determine the 'better' of two pathing algorithms to the same location based on parameters of safety vs speed vs distance, for example.

Learning Agent:
A learning agent is again built atop all other agent designs but crucially adds three new systems; a learning element, a critic, and a problem generator. The basic decision-making agent becomes a performance agent, which still chooses which actions to enact. The learning agent takes critiques from the critic, an unbiased unchanging scorer of the agent's actions, and attempts to modify the agent with new actions to take. This learning element can also take advice from the problem generator which tests the bounds of what the agent thinks is right and suggests some unoptimal steps to see what happens and learn from those random attempts.

2. Both the performance measure and the utility function measure how well an agent is doing. Explain the difference between the two.
[1 point]
The performance measure of an agent guides it to complete a task by somehow scoring its actions, making the agent want to go for the highest (or lowest) score it can achieve in its current state. The utility function looks at the various ways an AI solved a problem and weighs these solutions - generated by the AI using its performance measure - based on various parameters to determine the 'best' of these solutions.

3. In this question we explore further the differences between agent functions and agent programs. [2 points]

(a) Can there be more than one agent program that implements a given agent function?

Yes! An agent function guides the idea behind inputs and outputs, its just a map from point A to point Z. There can be many different agent programs that can take any number of steps to make these mapping pairs effective.

(b) Are there agent functions that cannot be implemented by any agent program?

Since agent functions can be as strict as the person who designs it wants it to be, the agent function may describe some task or vague event that is unimplementable. The classic halting problem, for example, can be written as a function but impossible to program.

(c) Given a fixed machine architecture, does each agent program implement exactly one agent function?

Yes. If there were some agent program that implemented more than one agent function it would end up trying miltuple different actions based on the same imput and most likely not complete either task or only focus on one and ignore the other.

(d) Given an architecture with *n* bits of storage, how many possible agent programs are there? Is this enough, that is, might there be environments for which there are no good agent programs?

Since an agent programs can be written in any number of ways to accomplish the same agent function, we can conclude there can be 2^n possible agent programs given n bits of storage. However, an infinite amount of storage (however high you can make n) can not alone make good agent programs, for example programs can have infinite storage but can still run forever if not made properly.

4. **Let us examine the rationality of various vacuum-cleaner agent functions in various environments.**
   [2 points]

   (a) Describe a rational agent function for the modified performance measure that deducts one point for each movement. Does the corresponding agent program require internal state?

   No because this performance measure does not require the agent to know its history. Deducting one point every time it moves (due to not seeing dirt) can be done at the same time as the move itself. There is no need to remeber to compuote performance measure this way.

   (b) Discuss possible agent designs for the cases in which clean squares can become dirty and the geography of the environment is unknown. Does it make sense for the agent to learn from its experience in these cases? If so, what should it learn?

   Yes! Primarily since the environment is unknown, it would be helpful to the agent to learn and map out its environment as it moves around so it makes less mistakes, like running into walls. It can also prioritize going to squares that have not just been cleaned over squares it just passed over depending on the frequency of clean squares becoming dirty.

Exercise 1.2 (10 points) The following exercises all concern the implementation of environments and agents for the vacuum-cleaner world. Programs need to be written in C/C++, Java or Python.

1. **Implement a performance-measuring environment simulator A *B* for the vacuum-cleaner world depicted in the figure on the right (Figure 2.2 on page 36 of our textbook). Your implementation should be modular so that the sensors, actuators, and environment characteristics (size, shape, dirt placement, etc.) can be changed easily. (Note: for CSC545/ECE537 students only: for some choices of programming languages and operating systems, this step can be skipped because there are already implementations in the online code repository.) [5 points if done yourself, none if code from AIMA is used. CSC645 students have to write the program themselves.]**

2. **Implement a simple reflex agent for the vacuum environment in Exercise 1.2.1. Run the environment simulator with this agent for all possible initial dirt configurations and agent locations. Record the agent's performance score for each configuration and its overall average score.**
   [5 points]