CISS450: Artificial Intelligence
Assignment 3

OBJECTIVES

- Understand the overall structure of various agents through software design.
- Write python code to implement various agents.

Several assignments will use graphical animation. I'll be using the pygame library. To install pygame, as user, in your bash shell do

```
pip install pygame
```

For this assignment, `pygame` is required. This should be installed in the fedora virtual machines we are using. If not, execute the following (as user):

```
pip install pygame
```

You are given a folder (see `agent`) with some code. In your bash shell execute

```
python main.py
```

or

```
make
```

Here's a sample execution:

```
[student@localhost agent]$ python main.py
random seed: 0
number of time steps: 3
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random): 0
location of agent (A or B): A
name of agent (return to end): c3p0
type of brain (0-random): 0
location of agent (A or B): B
name of agent (return to end):
graphical animation? (y/n): y

environment: clock:0, A:Dirty, B:Dirty, r2d2:A, c3p0:B

environment: clock:1, A:Dirty, B:Dirty, r2d2:A, c3p0:B
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'A'
action     : Suck
environment: clock:1, A:Clean, B:Dirty, r2d2:A, c3p0:B
running agent c3p0 ...
percept    : 'room_status': 'Dirty', 'location': 'B'
action     : Left
environment: clock:1, A:Clean, B:Dirty, r2d2:A, c3p0:A

environment: clock:2, A:Clean, B:Dirty, r2d2:A, c3p0:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Suck
environment: clock:2, A:Clean, B:Dirty, r2d2:A, c3p0:A
running agent c3p0 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Left
environment: clock:2, A:Clean, B:Dirty, r2d2:A, c3p0:A
```

```
environment: clock:3, A:Clean, B:Dirty, r2d2:A, c3p0:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:3, A:Clean, B:Dirty, r2d2:B, c3p0:A
running agent c3p0 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Suck
environment: clock:3, A:Clean, B:Dirty, r2d2:B, c3p0:A

simulator ended
press enter to end ...
```

Study the output and then study the code in `agent` carefully.

It will save you time if you enter inputs into a text file and run the program using I/O indirection. I have created one for you – see text file `stdin1.txt` and then in your bash shell execute

```
python main.py < stdin1.py
```

You can do the same if you do this:

```
make 1
```

Look at the `makefile`.

Q1. Make a copy of the given code and write an AI simulation where the agent is the simple reflex Vacuum Cleaner Robot agent in the AIMA textbook.

Your code should include a class called `SimpleReflexVacuumCleanerRobotBrain` in the file `SimpleReflexVacuumCleanerRobotBrain.py`. The brain should think like this (i.e., it should return the action as follows):

```
if the rooom is dirty:
    return Suck
else if the room is A:
    return Right
else:
    return Left
```

(This is similar to the reflex agent in AIMA p.48, Figure 2.8)

Modify `main.py` so that you can choose between two brains, the given random brain and the one you're building for this question.

Refer to the test cases below. Notice that you now can choose from two brains.

TEST 1.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Clean
status of room B (Clean or Dirty): Clean
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): A
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Clean, B:Clean, r2d2:A

environment: clock:1, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:1, A:Clean, B:Clean, r2d2:B

environment: clock:2, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:2, A:Clean, B:Clean, r2d2:A

environment: clock:3, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:3, A:Clean, B:Clean, r2d2:B

environment: clock:4, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:4, A:Clean, B:Clean, r2d2:A

environment: clock:5, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:5, A:Clean, B:Clean, r2d2:B

simulator ended
```

TEST 2.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Clean
status of room B (Clean or Dirty): Clean
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): B
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Clean, B:Clean, r2d2:B

environment: clock:1, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:1, A:Clean, B:Clean, r2d2:A

environment: clock:2, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:2, A:Clean, B:Clean, r2d2:B

environment: clock:3, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:3, A:Clean, B:Clean, r2d2:A

environment: clock:4, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:4, A:Clean, B:Clean, r2d2:B

environment: clock:5, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:5, A:Clean, B:Clean, r2d2:A

simulator ended
```

TEST 3.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Clean
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): A
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Dirty, B:Clean, r2d2:A

environment: clock:1, A:Dirty, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'A'
action     : Suck
environment: clock:1, A:Clean, B:Clean, r2d2:A

environment: clock:2, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:2, A:Clean, B:Clean, r2d2:B

environment: clock:3, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:3, A:Clean, B:Clean, r2d2:A

environment: clock:4, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:4, A:Clean, B:Clean, r2d2:B

environment: clock:5, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:5, A:Clean, B:Clean, r2d2:A

simulator ended
```

TEST 4.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Clean
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): B
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Dirty, B:Clean, r2d2:B

environment: clock:1, A:Dirty, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:1, A:Dirty, B:Clean, r2d2:A

environment: clock:2, A:Dirty, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'A'
action     : Suck
environment: clock:2, A:Clean, B:Clean, r2d2:A

environment: clock:3, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:3, A:Clean, B:Clean, r2d2:B

environment: clock:4, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:4, A:Clean, B:Clean, r2d2:A

environment: clock:5, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:5, A:Clean, B:Clean, r2d2:B

simulator ended
```

TEST 5.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Clean
status of room B (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): A
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Clean, B:Dirty, r2d2:A

environment: clock:1, A:Clean, B:Dirty, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:1, A:Clean, B:Dirty, r2d2:B

environment: clock:2, A:Clean, B:Dirty, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'B'
action     : Suck
environment: clock:2, A:Clean, B:Clean, r2d2:B

environment: clock:3, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:3, A:Clean, B:Clean, r2d2:A

environment: clock:4, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:4, A:Clean, B:Clean, r2d2:B

environment: clock:5, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:5, A:Clean, B:Clean, r2d2:A

simulator ended
```

TEST 6.

```
random seed: 0
number of time steps: 5
status of room A (Clean or Dirty): Clean
status of room B (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): B
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Clean, B:Dirty, r2d2:B

environment: clock:1, A:Clean, B:Dirty, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'B'
action     : Suck
environment: clock:1, A:Clean, B:Clean, r2d2:B

environment: clock:2, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:2, A:Clean, B:Clean, r2d2:A

environment: clock:3, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:3, A:Clean, B:Clean, r2d2:B

environment: clock:4, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:4, A:Clean, B:Clean, r2d2:A

environment: clock:5, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:5, A:Clean, B:Clean, r2d2:B

simulator ended
```

TEST 7.

```
random seed: 0
number of time steps: 6
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): A
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Dirty, B:Dirty, r2d2:A

environment: clock:1, A:Dirty, B:Dirty, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'A'
action     : Suck
environment: clock:1, A:Clean, B:Dirty, r2d2:A

environment: clock:2, A:Clean, B:Dirty, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:2, A:Clean, B:Dirty, r2d2:B

environment: clock:3, A:Clean, B:Dirty, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'B'
action     : Suck
environment: clock:3, A:Clean, B:Clean, r2d2:B

environment: clock:4, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:4, A:Clean, B:Clean, r2d2:A

environment: clock:5, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:5, A:Clean, B:Clean, r2d2:B

environment: clock:6, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:6, A:Clean, B:Clean, r2d2:A
```

```
simulator ended
```

TEST 8.

```
random seed: 0
number of time steps: 7
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 1
location of agent (A or B): A
name of agent (return to end):
graphical animation? (y/n): n

environment: clock:0, A:Dirty, B:Dirty, r2d2:A

environment: clock:1, A:Dirty, B:Dirty, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'A'
action     : Suck
environment: clock:1, A:Clean, B:Dirty, r2d2:A

environment: clock:2, A:Clean, B:Dirty, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:2, A:Clean, B:Dirty, r2d2:B

environment: clock:3, A:Clean, B:Dirty, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Dirty', 'location': 'B'
action     : Suck
environment: clock:3, A:Clean, B:Clean, r2d2:B

environment: clock:4, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:4, A:Clean, B:Clean, r2d2:A

environment: clock:5, A:Clean, B:Clean, r2d2:A
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:5, A:Clean, B:Clean, r2d2:B

environment: clock:6, A:Clean, B:Clean, r2d2:B
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'B'
action     : Left
environment: clock:6, A:Clean, B:Clean, r2d2:A

environment: clock:7, A:Clean, B:Clean, r2d2:A
```

```
running agent r2d2 ...
percept    : 'room_status': 'Clean', 'location': 'A'
action     : Right
environment: clock:7, A:Clean, B:Clean, r2d2:B

simulator ended
```

Q2. Now modify the simulation so that there are 4 rooms, A, B, C, D, arranged in a straight line:

Of course you will need to set the status of rooms A, B, C, D and also allow agents to be placed in A, B, C, or D. The following is part of the execution of a test case:

```
[student@localhost answer]$ python main.py
random seed: 5
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
status of room C (Clean or Dirty): Dirty
status of room D (Clean or Dirty): Dirty
number of agents: 2
name of agent (return to end): r2d2
type of brain (0-random, 1-simple): 0
location of agent (A or B or C or D): C
name of agent (enter to end): c3p0
type of brain (0-random, 1-simple): 1
location of agent (A or B or C or D): D
```

I hope you see what will go wrong with your agent from the previous question.

Q3. This is a continuation of the previous question.

Now modify the simulation so that your agent has a memory attribute. This attribute must be placed in a brain class with the name of

`ModelBasedReflexVacuumCleanerRobotBrain`

and the python file containing the brain must be

`ModelBasedReflexVacuumCleanerRobotBrain.py`

The name of this attribute must be `i` and it must be an integer variable – and no more. It must be initialized to 0. In the corresponding brain object, you can use `i`: you can store an integer value in `i` and you can read the value stored in `i`. DO NOT ADD ANY OTHER ATTRIBUTE IN THIS BRAIN CLASS. You will need to figure out how to use `i` to get the robot to clean up all the rooms.

(Technically speaking a model-based reflex agent also contains a model which is knowledge on "how things work in the world" – refer to your notes.)

Your model-based agent must ensure that all rooms are clean when given enough time – in at most 10 time steps.

To standardize output, here's the first part of a test case:

```
random seed: 0
number of time steps: 10
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
status of room C (Clean or Dirty): Dirty
status of room D (Clean or Dirty): Dirty
name of agent (return to end): r2d2
type of brain (0-random, 1-simple, 2-model): 2
location of agent (A or B or C or D): A
name of agent (return to end):
graphical animation? (y/n): n
```

NOTE. A couple of asides:

- What is the least amount of time – in terms of simulation time steps – you need to guanrantee that the agent cleans up all the rooms?
- If `Suck` takes up more energy then `Left` and `Right`, say `Suck` has a cost of 5 and `Left`

and `Right` has a cost of 1, will your agent minimize cost?

Q4. [IGNORE THIS QUESTION. THIS IS OPTIONAL.]

Now modify the simulation so that your agent is goal-based. The brain must be called

`GoalBasedVacuumRobotBrain`

and the python file containing the brain must be

`GoalBasedVacuumRobotBrain.py`

It has the same capabilities as the model based reflex agent (see previous question). The only difference is that the agent now has two extra actions: the agent can go into sleep mode (to save energy) or wake mode. Initially the agent is in sleep mode. To wake up, the agent execute `Wake` as an action. If the agent is in the wake mode, it can go into the sleep mode by execution the `Sleep` action. Of course if the agent is already in sleep mode, executing `Sleep` will not change its sleep mode. Likewise if the agent is already in wake mode, executing `Wake` will not change anything. Your agent must ensure that, when given enough time, it will clean up all the rooms and enter sleep mode.

Of course, if the agent is in `Sleep` mode, it cannot execute `Left`, `Right`, or `Suck`.

To standardize output, here's the first part of a test case:

```
[student@localhost agent]$ python main.py
status of room A (Clean or Dirty): Dirty
status of room B (Clean or Dirty): Dirty
status of room C (Clean or Dirty): Clean
status of room D (Clean or Dirty): Dirty
name of agent #0 (enter to end): r2d2
type of brain (0-random, 1-simple, 2-model, 3-goal): 3
```

The following standardizes the display of the environment:

```
environment: clock:2, A:Dirty, B:Dirty, r2d2:(A, Sleep), c3p0:(B, Wake)
```

Note that the state of the agent include its location and its `Sleep/Wake` state.

[Note that the `Sleep/Wake` states are more like internal states, i.e., they are not the agent's states with respect to the environment. However, to simplify the problem, we will view the `Sleep/Wake` state as part of the environment.]