JP2 2019: Lab exam (Version C)

This lab exam is intended to be completed during the 3-5pm Monday timeslot (Lab sections 5 and 6). If you are taking the exam at a different time, you should locate the correct specification instead of this one.

Overview

In this lab, you will write classes designed to solve a simple planning problem. You will define an **Agent** which can **pick up** and **put down** objects. An agent can only pick up one object at a time, so if it is already holding something, it needs to put it down before picking up the next object. Also, the agent is not reliable, so it might succeed or fail when it attempts to do one of those actions. You will then define a problem involving a number of objects and must choose a correct sequence of pick-up and put-down actions to move all of the objects to a defined location.

Here is an example: there are two locations in the world, the **Floor** and the **Table**, and three objects as follows:

- Object O1 is on the Table
- Objects O2 and O3 are on the Floor

Then, if an agent must solve this problem of moving all objects to the table, one possible plan is as follows (note that O1 does not need to move):

- Pick up O2
- Put O2 on the Table
- Pick up O3
- Put O3 on the Table

Since the agent is unreliable, you can't just execute the plan as above – you will need to check after each step whether the action succeeds. If an action is not successful, you will need to keep trying it until it is successful. So the actual sequence of actions might look more like this.

- Attempt to pick up O2 (fail)
- Try again to pick up O2 (fail)
- Try again to pick up O2 (success)
- Try to put down O2 on the Table (success)
- Try to pick up O3 (success)
- Try to put down O3 (fail)
- Try to put down O3 (success)

Task 1: Location (2 marks)

Note about implementation: all classes created in this exam should be put in the ***planning*** *package.*

You must create an enumerated type **Location** with the following values:

TABLE, FLOOR, HOLDING

Task 2: Thing (6 marks)

You must create a class **Thing** representing an object in the planning domain.[1] This class should have the following properties:

- name: (a String)
- location: (an object of type Location)

The **Thing** class should have a constructor with the following signature to set the values of its fields:

public Thing(String name, Location location)

The **Thing** class should also include the following:

- A complete set of **get** methods, and a **set** method only for the location
- Appropriate implementations of **equals()**, **hashCode()**, and **toString()**

Task 3: Agent (6 marks)

Create a class **Agent** representing an agent. This class should have a single field of type **Thing**, representing the object which the agent is holding, which should be initially set to **null**. **You do not need to override the equals(), hashCode(), or toString() methods for this class**.

In addition, **Agent** must provide the following two methods:

- **public boolean pickUp(Thing thing)**
- **public boolean putDown(Location place)**

The behaviour of these methods should be as follows:

pickUp

- If the agent is already holding something, return false immediately.
- Otherwise, choose a random Boolean value to determine whether the pick-up succeeds
  - If the random value is true, set the location of the **Thing** to **HOLDING**, update the "holding" field to be the given Thing, and return true
  - If the random value is false, do nothing and return false

[1] I didn't call this class **Object** because that would cause a lot of confusion with the built-in **java.lang.Object** class.

putDown

- If the agent is not holding anything, return false immediately
- If **place** is **HOLDING**, return false immediately
- Otherwise, choose a random Boolean value to determine whether the action succeeds
    - If the value is true, set the location of the thing that we were holding to **place**, set the "holding" field to null, and return true
    - If the value is false, do nothing and return false

Task 4: planning (5 marks)

Add an additional method to the **Agent** class with the following signature:

public void putOnTable(Set<Thing> things)

This method should process the given set of things to ensure that they are all located on the table at the end, using the **pickUp()** and **putDown()** methods as appropriate. Don't forget to check the return values to ensure that each action is successful.

What to submit

On Moodle, go to **Lab Exam Submission – Version C** and upload the following three files:

- Location.java
- Thing.java
- Agent.java

Be sure to submit to the correct assignment link, and be sure to submit before the deadline.