Individual assignment 2: Q-Learning

The goal of this assignment is to implement the Q-Learning algorithm as explained in the book of Ertel. However, instead of generating random paths yourself, you will be given a set of paths. This implementation should also be able to return the policy derived from the result of this algorithm.
This assignment has to be done in Java. A template Netbeans project for this can be found on the site.

Template

In this template you will find two classes: *QLearner* and *QLearnerTest*.

In the class *QLearner* you will implement Q-Learning in the method *execute*. This method has 4 parameters: *rewards*: the reward matrix; *paths*: an array of paths to go along; *gamma*: the learning rate γ; And *iterations*: the number of times the paths have to be traversed. The method should perform Q-learning by handling the given paths in order and should do so for the given number of iterations. The method should compute and eventually return a policy string. Furthermore,

- a **state** is an integer;
- an **action** is of the form: 'move to state s'; so, it can also be represented by an integer, namely s;
- a **path** $paths[i]$ is an array of integers. The first integer is the start state, the following integers are actions. So, if p is a path, then that path starts in $p[0]$ and the first action is move to state $p[1]$, etcetera;
- for the *rewards* matrix holds: $rewards[s][a]$ = the integer reward when applying action a in state s. If $rewards[s][a]$ equals *null*, action a can not be applied in state s;
- a **policy** gives an optimal action for every state.

As a string representation of a policy we use a space separated list of preferred actions. For instance `3 2 0 1` represents the policy that in the states 0, 1, 2, and 3, the preferred actions are 3, 2, 0, and 1, respectively. For a final state, a state without any possible action, write 'n' as the preferred action. For instance, the output "3 2 n 1" means that there is no action possible in state 2. If there are multiple optimal actions in a state, either one of them suffices.

In the class *QLearnerTest* you will be able to test your implementation using JUnit. There is an example test given to show how such a test is constructed. To execute the tests, select the Test menu item of the project.

Submission

The class *QLearner* with your implementation has to be handed in on Peach. Before you hand it in, please make sure that

- the file does not contain a package statement and
- there are no print statements in the file.