Homework 2

1. A Mars rover (robot) wants to go up onto a hill to charge itself through its solar panel. The robot has two different positions on the hill: **top** and **bottom**. The robot has two actions: "**drive"** and "**stop"**. There are two units of energy cost in taking the action "drive".

   When the robot is on top of the hill, it collects three units of energy. At the bottom, it collects one unit of energy.

   When the robot "drives" at the "top" position, there is 0.8 probability that it remains on top of the hill, while there's 0.2 probability of falling to "bottom". When the robot "stops" on top of the hill, the two probabilities are 0.6 and 0.4 respectively.

   When the robot drives at the bottom, there is 0.7 probability of getting to the top, and otherwise it remains at the bottom. When it stops at the bottom, it remains there.

   a. Define the problem as an MDP

   b. Draw a transition diagram to completely capture the MDP problem

   c. Under 0.8 discount factor, use the Value Iteration (VI) algorithm to compute value function after one iteration and two iterations: $V_1(s)$, and $V_2(s)$. Let's start with $V_0(s) = 0$.

   d. Given a policy π that always suggests the robot to drive, compute $V^{\pi}{}_1(s)$ and $V^{\pi}{}_2(s)$. The robot start from $V^{\pi}{}_0(s) = 0$.

   e. This is a follow-up question of Q-1d. Extract a new policy from $V^{\pi}{}_2(s)$.

2. The following shows a program in Answer Set Programming (ASP) to solve the map coloring problem. The “arc” predicate defines the connections between countries. The rule in the box is a “constraint”: the body (what’s on the right of :-) cannot be true.

   One of the solutions of this map coloring problem looks like the following, while there are totally more than 100 solutions.

   “color(belgium,red) color(denmark,red) color(germany,green) color(netherlands,blue) color(luxembourg,blue) color(france,white)”

   You can install Clingo to verify your solution. To install Clingo on Ubuntu:

   sudo apt-get install gringo

   To test your code (optionally adding “-n 0” to the end gives all the solutions):

   clingo coloring.lp4

```
countries(belgium;denmark;france;germany;netherlands;luxembourg).

colors(red;green;blue;white).

arc(france,belgium).
arc(france,luxembourg).
arc(france,germany).
arc(luxembourg,germany).
arc(luxembourg,belgium).
arc(netherlands,belgium).
arc(germany,belgium).
arc(germany,netherlands).
arc(germany,denmark).

neighbour(X,Y) :- arc(X,Y).
neighbour(Y,X) :- arc(X,Y).

Blank 1 :- color(X, C1), countries(X), colors(C1), colors(C2), C1!=C2.
color(X, C) :- countries(X), colors(C), not Blank 2 .

:- color(X1, C), color(X2, C), neighbour(X1,X2).

% symmetry breaking
:- color(germany, red).

#show color/2.
```

   a. Fill in Blank 1, and explain what the rule in which Blank 1 is means.
   b. Fill in Blank 2, and explain what the rule in which Blank 2 is means. The “not” symbol means default negation.
   c. Describe the consequence of removing the two “neighbour” rules right above Blank 1.