DPRL Assignment 2

- Implement the following exercise in python using standard functions and packages (no MDC packages)
- We consider a problem with the following features and parameters:
  - A system consists of 2 components that both need to be functioning for the whole system to work
  - Each component can be in any of 10 states indicating the level of deterioration; state 10 = failure
  - Every time unit there is a 10% probability for each component that it deteriorates to the next state
  - A reward of 1 is received when the system is functioning and no component is being repaired
  - There are two types of repair: preventive and corrective, the latter happens (immediately) when one of the components has failed
  - Repair takes 1 time unit, afterwards the component(s) that is/are repaired is in state 1. If 1 component is being repaired, the other one does not deteriorate
  - Preventive repair costs 5 per component, corrective repair 25 per component
  - It is possible to repair both components at the same time, preventing additional downtime
  - The objective is the maximize the long-run average reward

a) Give all components of the MDP: states, actions, transitions, rewards.

b) For the system without preventive repair, compute the long-run expected reward in 3 ways:
   1. Simulate for 1000000 time units and give the average reward;
   2. By determining the stationary distribution in a forward recursive manner;
   3. By solving the Poisson equation using value iteration.

c) Now allow for preventive repair of 1 component if the other has failed. Determine using value iteration the optimal policy. Give its value and plot the optimal policy. Interpret your results.

d) Now allow for preventive repair of 1 or 2 components in any state. Determine using value iteration the optimal policy. Give its value and plot the optimal policy. Interpret your results.

How and what to submit

- A report (.pdf) of max 2 A4 pages plus appendix with relevant figures/tables/screenshots OR max 1000 words
- A separate file with Python code
- Implement the algorithm in an efficient way, it should run very fast
- Report should include the solution method. Mathematically describe the method that you coded, including implementation choices and initialization and stopping criteria of the method
- Comment on your findings, are they as expected?
- Grading:
  - 1 + a + 2b.1 + 2b.2 + 2b.3 + c + d