(1) This problem is called the 687 grid world problem. We have a 5 x 5 grid grid with grid positions (i, j), $0 \leq i, j \leq 4$. Both $(2, 2)$ and $(3, 2)$ are obstacles and a robot cannot enter these positions. The goal state is $(4, 4)$ and state $(4, 2)$ has water. The robot starts at $(0, 0)$ and would like to reach the goal state. Actions are attempting up, down, left and right. With probability 0.8 a robot moves moves in the direction it attempts. With probability 0.05 it moves at 90 degrees to the right of what it attempts and probability 0.05 it moves at 90 degrees to the left of its attempted direction and stays back with the remaining probability. So, if it were attempting right move at $(1, 1)$ it will go to the right with probability 0.8, down with probability 0.05, up with probability 0.05 and stay back with probability 0.1. If the dynamics causes it to get out of the grid it remains in the current position with the probability which caused it to get out of the grid. If the dynamics causes it to get into $(2, 2)$ and $(3, 2)$ it remains at the state it is in with the appropriate probability. The agent receives -10 for entering a water state and +10 for getting to the goal. If it is in $(4, 2)$ and the next move causes it to remain there, it receives -10 again. Use discount parameter 0.9.

i Implement a parametrized tabular policy search algorithm with parameter space θ^n, $n = |\mathcal{S}||\mathcal{A}|$ using gradient ascent.

For a hyperparameter σ, $\pi(a|s; \theta) := \frac{exp^{\sigma\theta_{s,a}}}{\sum_{a'} exp^{\sigma\theta_{s,a'}}}$ is the probability of selecting action a. You wish to find θ maximizing the expected return. To do this start with a mean policy parameter vector $\theta \in \mathbb{R}^n$. Note that by fixing θ, σ, we get a policy. Evaluate the policy by running N episodes of the same and calculating the average gain G, (try $N = 10, 20$ (maybe more, if the results are not good). Select θ' from a normal distribution $\mathcal{N}(\theta, \sigma I)$. Again evaluate the gain G' for this choice of θ', and if $G' > G$ set $\theta = \theta'$, $G := G'$, otherwise pick a new θ' (this is called Hill search/ Gradient ascent). Repeat the process with θ' instead of θ - till the difference G', G is less than a chosen ϵ. State how you searched the hyperparameter space. Plot a learning curve of the expected gain versus episodes using the best hyperparameters. For the plot you will need to average the following curves - each curve corresponds to a trial where we run the algorithm for a large number of episodes. The curve will be a plot of episode number and gain in that episode. We will average such curves over say 300 trials. Draw the standard error bars too.

i Implement value iteration and obtain an optimal policy. How many iterations does this take before converging? For the optimal policy draw curves as above. Plot the two curves you get using policies in $(i), (ii)$ on the same graph. How do the two policies compare?

(2) The dynamics of the cart-pole balancing is discussed in detail on a paper uploaded on Moodle. Use the calculation discussed in that paper assuming you are in the frictionless case and assume $g = 9.8$. The RL agent must learn to move the cart forwards and backwards and prevent the pole from falling. The state is a tuple $s = (x, v, \theta, \dot{\theta})$ x being the horizontal position of the cart, θ the angle of the pole w.r.t the vertical, v the velocity of the cart and $\dot{\theta}$ being angular velocity. To make this a finite horizon MDP you can include time in a state so a state is (x, t) - increment t by 1/50-seconds and, provided the pole has not fallen you can find the new state for this time. We assume the x coordinate runs from $[-3, 3]$. Assume failure occurs when the angle is not in the range $[-5\pi/12, 5\pi/12]$ (you may need to modify this). We always start with state $(0, 0, 0, 0, 0)$.

Terminate an episode if the pole falls, or if cart-pole hits or crosses a boundary point or the time is 20 seconds.

For the dynamics you can assume $F = 10N$, cart mass m_c is 1Kg, pole mass m_p is 0.1 kg. You need to bound the cart velocity in the range $[-10, 10]$ so if the equations computed result in velocity less than -10m/s then the velocity is set to -10m/s and likewise if it exceeds 10m/s. The angular velocity can similarly assumed to be bounded in the range $[-\pi, \pi]$. There are two actions, left and right. Assume that the discount factor is 1 and the reward at each time step is 1. So the total reward is the time until which the pole falls.

i Implement a parametrized gradient ascent algorithm for cart-pole. Now this is not a discrete space so you will have to choose an appropriate representation for the parametrized policy for example linear and/or quadratic functions in $x, v, \theta, \dot{\theta}$.
State how you searched the hyperparameter space.
Plot a learning curve of the expected gain versus episodes using the best hyperparameters. For the plot you will need to average the following curves - each curve corresponds to a trial where we run the algorithm a large number of episodes. The curve will be a plot of episode number and gain in that episode. We will average such curves over say 300 trials. Draw the standard error bars too.
Make sure your curves converge as the number of episodes increases to the optimal return. What is that number?

ii Also implement the cross entropy method on this problem. Recall for this you start by fixing an ϵ and fixing numbers K, K_ϵ. Select an initial θ and a covariance matrix $\sum := 2I$. Pick θ_k from the distribution $\mathcal{N}(\theta, \sum)$ for $k = 1, \dots, K$. Run $N = 10$ episodes of the policy with respect to each θ_k and compute the average gain over these episodes for each θ_k. Pick the top K_ϵ winners among the θ_k's and compute the average $\bar{\theta}$ of these K_ϵ winners. Assume that the top k are $k = 1, \dots, K_\epsilon$ and update $\sum$ via

$$\sum := \frac{1}{\epsilon + K_\epsilon}(\epsilon 2I + \sum_{k=1}^{k=K_\epsilon}(\theta_k - \bar{\theta})(\theta_k - \bar{\theta})^T)$$