PAGE
DATE

Reinforcement learning
assignment 1

* Exercise 1

1- the best expection since we know the Probabilties is individual estimation for cases A and B are not Possible in the first scenario. As a result, the ideal way is to choose the action with the best value estimate in combination. In this scenario, the estimates for both activities are identical. So the optimum chance of success is 0,5, which may be obtained by picking at action at random in each phase.

$A_1 = \sum x_i P(x_i) = 0,5 * 0,1 + 0,5 * 0,9 = 0,5$

$A_2 = 0,5 \times 0,2 + 0,5 * 0,8 = 0,5$

2- In the second scenario, we can hold indefendant estimates for cases A and B, allowing us to learn the appropriate action for each by considering them as distinct bandit Problems, we can map A_2 to case A and A_1 to case B to reach the maximum value of success.

$A_1 = 0,5 * 0,2 + 0,5 \times 0,9 = 0,55$

$A_1 = 1 - 0,55 = 0,45$

PAGE
DATE

Exercise 4

Since this is continuing task with discount rate, G_t here is bounded and the reward is always 1

$$G_t = R_{t+1} + \gamma G_t X_i$$

$$G_t = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma} \quad \text{for } 0 \leq \gamma \leq 1$$

If this value is getting higher then we can say that our agents are improving since the objective of RL is to maximizing the expected reward, and how good for the agent to be in a sta

PAGE
DATE

Exercise 3 :-

* Since we're tracking a non-stationary Problem, we can use a constant step size :

$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$ subject to $\alpha \in [0,1]$ is constant

$$
\begin{aligned}
Q_{n+1} &= Q_n + \alpha[R_n - Q_n] \\
&= \alpha R_n + (1-\alpha) Q_n \\
&= \alpha R_n + (1-\alpha)[\alpha R_{n-1} + (1-\alpha) Q_{n-1}] \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\
&= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \dots + (1-\alpha)^{n-1} \alpha R_1 + (1-\alpha)^n Q_1
\end{aligned}
$$

So $Q_n = (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha (1-\alpha)^{n-i} R_i$

Yes, It's an exponential weighted average, because α is between [0,1] and by that the sum of all weights over time equal to 1, and the expected reward affected by recent rewards more than old ones.

If we denote the rewards by vector **R**, the weighted average will be be taking the dot Product between **R** and vector **W** such that $0 \leq w_i \leq 1$ and the sum of all w_i is 1.

$$w_i = \frac{\alpha^i}{\sum w_i}$$

When the number of trials is 100 and the discount rate is 0.9

+ Code + Markdown

```
np.round(markovian((5,5),100,0.9), 1)
```

```
array([[ 3.3,  8.8,  4.4,  5.3,  1.5],
       [ 1.5,  3. ,  2.3,  1.9,  0.5],
       [ 0.1,  0.7,  0.7,  0.4, -0.4],
       [-1. , -0.4, -0.4, -0.6, -1.2],
       [-1.9, -1.3, -1.2, -1.4, -2. ]])
```

When the number of trials is 100 and the discount rate is 0.85

```
np.round(markovian((5,5),100,0.85), 1)
```

```
array([[ 2.9,  9.1,  4. ,  5.2,  1.2],
       [ 1.2,  2.7,  1.9,  1.6,  0.3],
       [-0.1,  0.6,  0.5,  0.3, -0.4],
       [-0.9, -0.4, -0.3, -0.5, -1. ],
       [-1.6, -1.1, -1. , -1.2, -1.7]])
```

When the number of trials is 100 and the discount rate is 0.75

```
np.round(markovian((5,5),100,0.75), 1)
```

```
array([[ 2.2,  9.4,  3.3,  5.1,  0.7],
       [ 0.7,  2.2,  1.3,  1.2,  0.1],
       [-0.2,  0.4,  0.3,  0.2, -0.4],
       [-0.7, -0.3, -0.2, -0.3, -0.8],
       [-1.3, -0.8, -0.7, -0.8, -1.3]])
```