Answer Sheet

1.

States:

Hostel (H): Reward = -1

Academic Building (A): Reward = +3

Canteen (C): Reward = +1

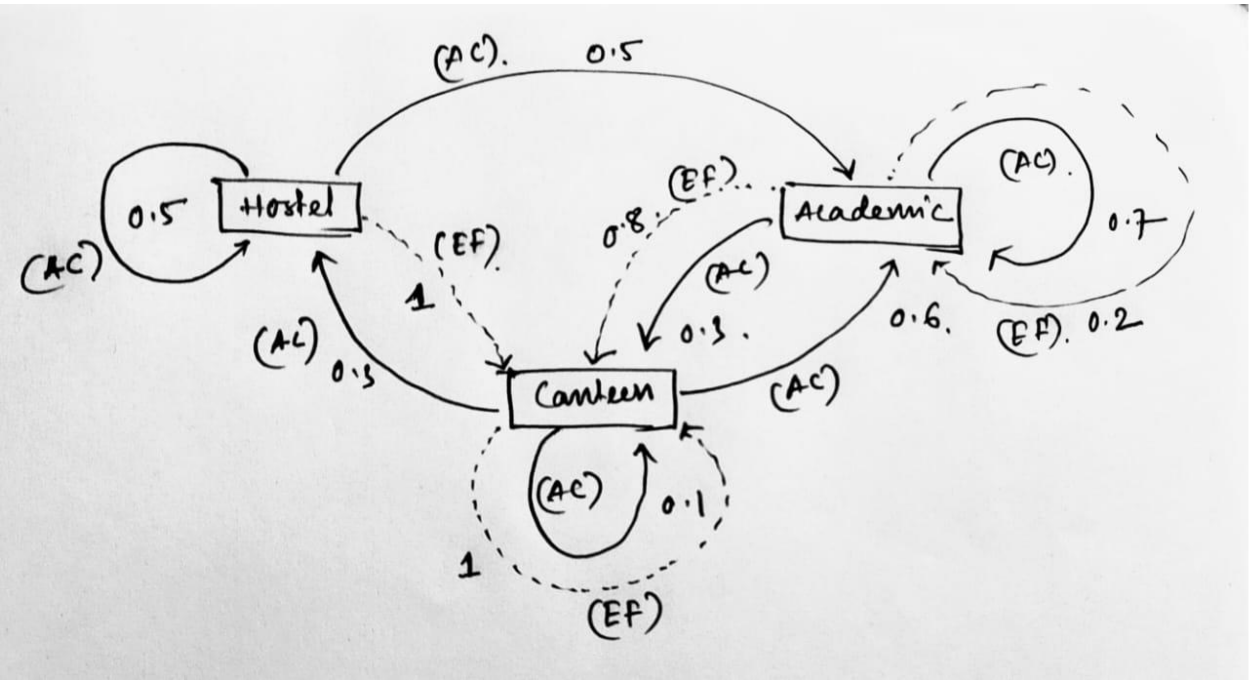
Actions:

Attend class (AC): The student attempts to attend a class.

Eat food (EF): The student goes to the canteen to eat food when hungry.

Current state (S)	**Action (A)**	**Next state (S')**	**Transition probability P(S'|S, A)**	**Reward R(S')**
Hostel (H)	Attend class	Academic (A)	0.5	+3
Hostel (H)	Attend class	Hostel (H)	0.5	-1
Hostel (H)	Eat food	Canteen (C)	1.0	+1
Academic (A)	Attend class	Academic (A)	0.7	+3
Academic (A)	Attend class	Canteen (C)	0.3	+1
Academic (A)	Eat food	Canteen (C)	0.8	+1
Academic (A)	Eat food	Academic (A)	0.2	+3
Canteen (C)	Attend class	Academic (A)	0.6	+3
Canteen (C)	Attend class	Hostel (H)	0.3	-1
Canteen (C)	Attend class	Canteen (C)	0.1	+1
Canteen (C)	Eat food	Canteen (C)	1.0	+1

MDP Diagram:



Value iteration and policy iteration in jupyter notebook, named with Qustion_1.ipynb

Comparison of results from policy iteration and value iteration:

1. Optimal Policies:

Both policy iteration and value iteration resulted in the same optimal policy for all states:

Hostel (H): Attend class, Academic Building (A): Attend class, Canteen (C): Attend class

This indicates that the best action, is for the student to prioritize attending classes over eating food.

2. Value function:

The optimal values for each state obtained from value iteration are:

V(H)=18.95

V(A)=20.94

V(C)=19.81

The values suggest that starting at the Academic Building (A) is the most favorable since it yields the highest expected cumulative reward, followed by the Canteen (C), and then the Hostel (H).

3. Convergence:

Value iteration: Converged after 138 iterations.

Policy iteration: Converged in 0 iterations (i.e., the initial policy was already optimal). Policy iteration typically converges faster than value iteration because it explicitly evaluates and improves policies rather than values.

However, both methods lead to the same optimal policy, but policy iteration did so with fewer iterations.