Name: Sadneya Samant
Roll No: 46
class: D15C

05/05

AIDS Assignment

Q.1. What is AI? Considering the COVID-19 pandemic situation, how did AI help to survive and renovate our way of life with different applications?

→ 1) Artificial Intelligence is the replication of human intelligence in machines, enabling them to learn, reason, solve problems, and makes decisions without direct human interaction.

2) AI played a vital role in managing the COVID-19 crisis by detecting outbreaks early through data analysis, accelerating drug and vaccine development, automating medical diagnoses with imaging and predictive models, supporting telemedicine via AI chatbots, optimizing supply chains for essential goods and enhancing remote work and education through AI-driven automation, making daily life more adaptable and efficient.

Q.2. What are AI agent terminology? Explain with examples.

→ AI agents are systems that perceive their environment, process information and take actions to acheive specific goals. Key terminologies include:

1) Agent - An entity that interacts with environment. (eg. self-driving car).

2) Environment - The externel system in which the agent operations (eg. traffic conditions for a self-driving car)

3) Perception - Data collection from sensors (eg. traffic conditions for a self-driving car)

4) Actuators - Components that execute actions (eg. robotic arms in manufacturing)

5) Rationality - The ability to make optimal decisions based on available information (eg. AI trading bots adjusting stock investments)

6) Autonomy - The degree of independence an agent has (eg. voice assistants like Alexa operating without human input)

7) Types of agents:

1) Simple reflex agents: Act based on current perceptions (eg. Thermostats)

2) model-based agents: use internal models to predict future states. (eg. GPS navigation system)

3) Goal-based Agents: Take actions to achieve a specific goal. (eg. chess-playing AI).

4) Utility-Based agents - optimize outcomes for maximum benefit (eg. recommendation systems in e-commerce.)

5) Learning agents - Improve performance over time using past experiences (eg. self-learning robots in warehouses.)

Q.3. How is the AI technique used to solve 8-puzzle probleme

→ 1) In 8-puzzle technique it consist of 3x3 sized cube. In initial state it contains all number tiles present randomly.

2) Then Here, our aim is manage them in increasing order from 1 to 8. by moving tiles with the help of blank space.

3) Common AI techniques include:

i) uninformed search:

1) Here by using BFS we can explore all possible moves level by level but it is slow. it guarantess the shortest path.

2) Then by using DFS we can explore a path deeply before backtracking. it maay not always find optimal solution.

3) Then by Using IDDFS which combines DFS and BFS to find the shortest solution efficiently.

ii) Informed search: Techniques (Heuristic search)

1) Uses a heuristic function $f(n) = g(n) + h(n)$, where:
$g(n)$ is the cost to reach the current state.
$h(n)$ is the estimate cost to reach the goal.
2) The by using Greedy Best-first search we can find optimal solution which can be fast but not always optimal. It uses heuristic function $h(n)$
Here A* algorithm is the most efficient and widely used technique for solving 8-puzzle problem.

Q.4. What is PEAS descriptor?
PEAS (Performance measure, Environment, Actuators, Sensors) is a framework used to define the components of an AI agent by specifying how it interacts with its environment and evaluates its success.
PEAS description for given AI agents:
1) Taxi driver:
1. Performance Measure: Safety, speed, customer satisfaction, Fuel efficiency.
2. Environment: Roads, traffic, pedestrians, weather conditions.
3. Actuators: steering, acceleration, brakes, turn signals.
4. sensors: GPS, cameras, speedometer, LIDAR, fuel gauage.
2) Medical diagnosis system:
1. performance Measure: Accuracy of diagnosis, patient recovery rate, response time.
2. Environment: Patient data, medical records, symptoms, lab reports.
3. Actuators: Display diagnosis, presribe medication, recommend tests
4. Sensors: Patient input, test result, doctor's notes, medical imaging.
3) A music composer:

1. Performance measure: musical quality, originality, listner preference.
2. Environment: musical genres, user preferences, historical compositions.
3. Actuators: generating notes, melodies, instrument selection.
4. sensors: user feedback, music databases, emotional tone analysis.

4) An aircraft autolander

1. performance measure: safe landing, smooth touchdown, passenger comfort.
2. Environment: weather conditions, runway, air traffic, altitude
3. Actuators: Flaps, landing gear, brakes, enging thrust control.
4. sensors: Altimeter, GPS, wind speed sensors, radar, cameras

5) An Essay Evaluator

1. Performance Measure: Grammer accuracy, coherence, relevance, readibility.
2. Environment: Essays, writing rules, predefined grading criteria.
3. Actuator: score assignment, grammer suggestions, feedback generation.
4. sensors: Text input, grammer and plagiarism checkers, semantic analyzers.

6) A robotic sentry gun for the keck lab

1. Performance Measure. Accuracy, target identification, security effectiveness.
2. Environment: lab perimeter, potential threats, lighting conditions
3. Actuators: Rotating turret, firing mechanism, alarm system.
4. sensors: Motion detectors, infrared sensors, cameras, radar

Q.5. categorize a shoping bot for an offline bookstore according to each of six dimensions (fully / partially observable, deterministic / stochastic, episodic / sequential, static / dynamic, discrete / continuous, single / multi agent).

→ 1) partially observable: as bot may not have full knowledge of stock availability, customer preferences, or real-time changes in the bookstore.

2) stochastic: Book availability, customer behaviour and price changes introduce randomness, making environment unpredictable.

3) sequential: Each decision affects future interactions.

4) Dynamic: The environment can change as books sell out, customers move or bookstore policies update.

5) Discrete: The bot operates with distinct choices, such as recommending a book, checking stock or processing a purchase.

6) multi-agent: as bot interacts with customers, bookstore staff and possibly other AI systems.

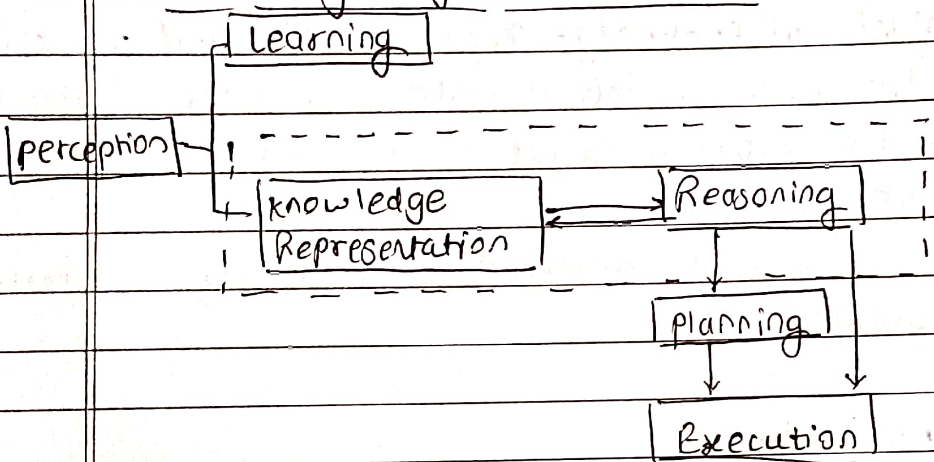
Q.6. Model based vs utility based agent.

	Model based	Utility-based
1)	used internal state to make decisions	choose actions based on maximizing a utility function
2)	predicts future states using the model before acting.	Evaluates multiple possible actions and selects best based on utility.
3)	It is goal oriented but focuses on how the environment works	It is also goal-oriented but it aims to achieve the best possible outcome.
4)	It may not always take the	Always selects the action that

best possible action, only takes a feasible one.	provides the highest benefit.
eg. self-driving car to predict congestion	eg. stock trading AI selecting the trade with the highest expected profit.

Q.7. Explain the architecture of knowledge based agent and learning agent.

→ Knowledge agent Architecture



1) perception : 1) it retreives data or information from the environment
2) you can retrieve data from the environment, find out noises and check if the AI was damaged by anything.
3) it defines how to respond when any sense has been detected.

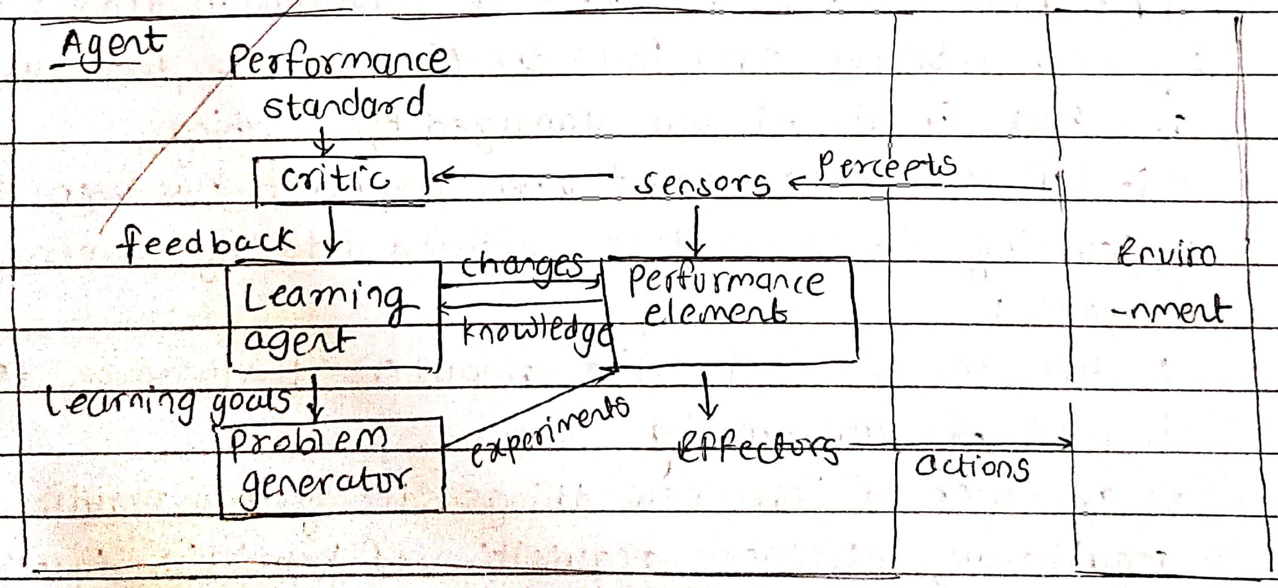
2] learning : 1) it learns from captured data by the perception component.
2) Here, the goal is to build computers that can be thought instead of programming them.
3) In order to learn new things, the system require knowledge acquisition, inference, acquisition of heuristics, faster searches, etc

3] knowledge representation & reasoning : 1) The main component in the cycle is knowledge representation and reasoning which

shows human-like intelligence in the machines.
2) Knowledge representation is all about understanding intelligence.
3) Goal is to understand and build intelligent behavior from the top-down and focus on what an agent needs to know in order to behave intelligently.
4) It defines how automated reasoning procedures can make this knowledge available as needed.

4] Planning and Execution: 1) The planning and execution components depend on analysis of knowledge representation and reasoning.
2) planning includes giving an initial state, finding their preconditions and effects, and a sequence of actions to achieve a state in which particular goal holds.
3) Now once the planning is completed, the final stage is execution of the entire process.

Learning Agent Architecture



FOR EDUCATIONAL USE

A learning agent improves its performance over time by learning from past experiences. It has four components:

1) Learning element: learns from interactions and updates knowledge.

2) Performance element: uses the learned knowledge to make decisions.

3) critic: evaluates agents performence by comparing actions with desired outcome.

4) problem generator: suggests new explanatory actions to improve learning.

Q.9 convert the following to predicates:

a) Anita travels by car if available otherwise travels by bus.

b) Bus goes via Andheri and Goregaon.

c) car has puncture so is not available. will anita travel via goregaon? use forward reasoning.

→ 1) Anita travels by car if ~~available otherwise~~ by bus.

so, Available (car) => Travels (Anita, car)

~Available (car) => Travels (Anita, Bus)

2) Bus goes via Andheri and Goregaon.

Route (Bus, Andheri)

Route (Bus, Goregaon).

3) car has a puncture, so it is not available.

Puncture (car) => ~ Available (car)

given: Puncture (car)

therefore, ~Available (car)

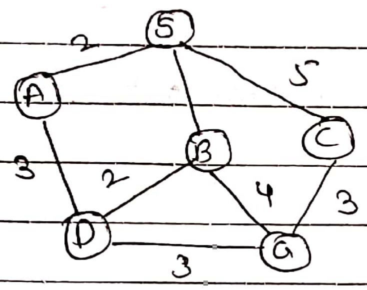
forward reasoning:

1) from (3) ~Available (car)

2) from (1), ~Available (car) => Travels (Anita, Bus)

so Anita must travel by bus.
3) from (2), Route(Bus, Goregan) meaning bus travels via goregaon.
4) since Anita travels by Bus and the Bus goes via Goregaon, Anita will travel via Goregaon.

Q.10. Find the route from S to G using BFS.



step 1: start with node S
mark it visited. | S |
eneque S into the queue
step 2: dequeue S and explores its neighbors A, B, C.
mark A, B, C visited.
enqueue A, B, C into the queue. | A | B | C |
step 3: deque A and explore its neighbours D.
mark D visited
enqueue D into the queue | A | B | C | D | ∴ S → A
step 4: deque B and explore its neighbours
mark G visited. | C | D | G | ∴ S → A → B
~~Thus goal is reached.~~
∴ ~~shortest path found will S → C → G~~
step 5: deque C and explore its neighbours
it has only 1 neighbour G which is already visited. ∴ S → A → B → C
step 6: deque D and explore its neighbours
it has only 1 neighbour G which is already marked visited
∴ S → A → B → C → D
step 7: deque G. thus goal state reached.
∴ S → A → B → C → D → G

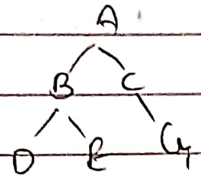
Q.11. What do you mean by depth limited search ? Explain iterative deepening search with examples.

→ 1) Depth - limited search is a variation of Depth-first search (DFS) where the search is restricted to a specific depth limit L.

2) if goal is not found within this limit, the search terminates.

3) it prevents infinite loops in deep or infinite state spaces but risks failing to find solution if L is too low.

eg.

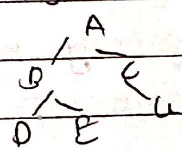


Here if L=1 and we need to find G then first expand A → [B, C] at L=1 thus goal ~~then for L=2 expand B→~~ not found so we need to increase the limit to 2 to provide solution. (this is main drawback.)

Iterative Deepning Search (IDS).

1) It combines the space efficiency of DFS and completeness of BFS by repeatedly running DLS with increasing depth limits L = 0,1,2,.. until goal is found

eg.



1) DLS with L=0 only A is present thus goal not found

2) increase L = 1

3) DLS with L=1 expand A, B and C found

4) again increase L = 2

5) DLS with L=2, Here Expand B gets D & E.
Expand C gives G thus goal found at depth = 2 (ie L=2)

2) thus it guarantees finding the shortest path while using less memory than BFS.

Q.12. Explore hill climbing and its drawbacks in detail with example. also state limitations of steepest - ascent hill climbing.

→ 1) Hill climbing is a local search algorithm that continuously moves toward the best neighbouring state with a higher heuristic value, aiming to reach a global optimum.

2) steps: 1) start from initial state.

2) evaluate neighbouring states and move to the one with highest heuristic value.

3) repeat until no better neighbour exists (local maximum).

eg. Imagine a mountain climbing scenario where a hiker moves uphill based on the steepest slope. if they reach a peak that is not the highest (global maximum), they might get stuck.

4) drawbacks of hill climbing:

i) local maxima: 1) the algorithm may stop at a peak that is not the global optimum.

2) eg. a small hill before taller mountain.

ii) Plateau problem: 1) A flat region where all neighbouring states have the same heuristic value, causing the search to halt.

2) eg. A long flat road with no clues on where to go.

iii) Ridges: The algorithm may fail to climb diagonally when only direct moves are considered. eg. climbing a staircase with missing steps.

iv) No backtracking: once it moves forward, it can't recover from a bad decision. eg. choose the wrong path in maze with no way back.

5) solution to improve hill climbing:

1) Random Restarts: start from different points.

2) Simulated Annealing: Occasionally accept worse moves to escape local optima.

Q.13. Explain simulated annealing and write its algorithm.

→ 1) simulated Annealing is an optimization algorithm designed to search for an optimal or near-optimal solution in a large solution space.

2) The name and concept are derived from the process of annealing in metallurgy, where a material is heated and then slowly cooled to remove defects and acheive a stable crystalline structure.

3) In simulated annealing, the "heat" corresponds to the degree of randomness in the search process, which decreases over time (cooling schedule) to refine the solution.

4) The method is widely used in combinational optimization, where problems often have numerous local optima that standard techniques like gradient descent might get stuck in.

5) Algorithm:

1) The algorithm starts with ~~an initial~~ solution and a high temperature, which gradually decreases over time. Here's step by step of how the algorithm works:

1) initialization: Begin with initial solution So and initial temperature To. the temperature controls how likely the algorithm is to accept worse solutions as it explores the search space.

2) Neighbourhood search: At each step, a new solution s' is generated by making a small change (or perturbation) to the current solution s.

3) objective function evaluation: The new solution s' is evaluat-ed using the objective function. If s' provides a better solution than s, it is accepted as the new solution.

4) Acceptance Probability: If s' worse than s, It may still be accepted with a probability based on temperature and the difference in objective function values. The acceptance probability is given by: $p(accept) = e^{-\Delta E/T}$

5) Cooling schedule: After each iteration, the temperature is decreased according to a predefined cooling schedule, which determines how quickly the algorithm converges. Common cooling schedules include linear, exponential or logarithmic cooling.

6) Termination: The algorithm continues until the system reaches a low temperature or a predetermined no. of iterations is reached.

Q.14. Explain A* algorithm with an example.

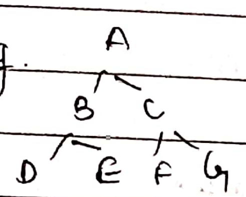
→ 1) The A* algorithm is an informed search algorithm that finds the optimal path by considering the both cost to reach a node ($g(n)$) and estimated cost to the goal ($h(n)$), using the formula: $f(n) = g(n) + h(n)$

where, $g(n)$ = actual cost from start node to n.
$h(n)$ = estimated cost from n to goal.

2) Algorithm:

1) Initialize the open list with start node.
2) loop until goal is found.
   1) pick the node with lowest $f(n)$.
   2) if its the goal node, return the path.
   3) expand its neighbors and calculate $f(n)$.
   4) Add unvisited neighbors to the open list.
3) continue until open list is empty.

3) eg.



Cost & Heuristic values

Node	(path cost) g(n)	(heuristic) h(n)	f(n)
A	0	6	6
B	1	4	5
C	2	3	5
D	3	7	10
E	4	5	9
F	3	2	5
G	4	0	4 (Goal)

start at A, calculate f(A) = 6
Expand A → {B, C} (both have f = 5)
Expand C → {F, G} (G have lowest f)
goal reached at G
∴ final path A → C → G

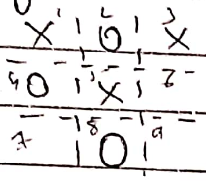
4) Advantages:
1) It is flexible hence used in maze solving, AI, robotics.
2) It is efficient as explores fewer nodes than BFS / DFS.
3) It provide guaranteed optimality.

5) Disadvantages:
1) It requires more memory than DFS.
2) A poor heuristic can make A* inefficient.

Q. 15. Explain minimax algorithm and draw game for Tic-tac-toe.
1) The minimax Algorithm is used in adversial search to determine the optimal move by assuming both players play optimally.
2) maximizer: (eg X) tries to get highest score.
minimizer: (eg O) tries to reduce the score.
3) Algorithm:
i) generate the game tree up to a depth limit.

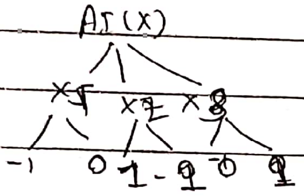
2] Assign heuristic values to leaf nodes.
3] Backpropogate values:
   1) maximizer picks the maximum value:
   2) minimizer picks the minimum value.
4] The root node selects the best move based on propogated values.

eg. 1) consider a Tic-Tac-Toe board where its AI's turn (x) to play:



so here possible moves can be done at position ~~...~~: (2,3), (3,1), (3,3).

2) The tree expands, considering all possible outcomes.



3) The AI (max) picks the more leading to (best possible score).

4) If multiple moves have the same values, additional heuristics (like the earliest win) may be used.

Q.16. Explain Alph-beta pruning algorithm for adversial search with an example.

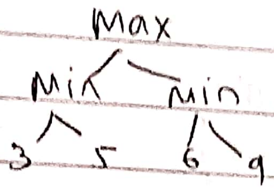
→ 1) Alpha-beta pruning optimizes minimax by skipping unnecessary branches, reducing computations. it uses two values:
Alph(α): Best score maximizer can achieve.
Beta(β): Best score minimizer can achive.

2) Algorithm:
1] perform minimax search.
2] Prune branches where:
   1) if Maximizer's best (α) $\geq$ Minimizer's best (β), further evaluation is skipped.
   2) if minimizer's best (β) $\leq$ Maximizer's best (α), further

evaluation is skipped.

3) This reduces time complexity without affecting final decision.

3) eg. let's consider a simple game tree with a depth of 2.



with α-β pruning,

~~left subtree evaluates~~

1) start at the root (MAX node)

$\alpha = -\infty \quad \beta = +\infty$

2) move to the first MIN node (left subtree).

3) evaluate the first child of MIN (3)

4) thus MIN updates $\beta = 3$ as it is the smallest value seen so far)

5) now move to second child of MIN (5).

$\therefore \beta = \min(5, 3) = 3$

6) move back to MAX. MAX updates $\alpha = \max(-\infty, 3) = 3$

7) Now move to second MIN node (right subtree)

8) Now $\alpha = 3$ (best MAX value so far) and $\beta = +\infty$

9) after evaluating first child of MIN (6) β gets equal to 6.

10) Now check pruning condition before evaluating 9:

$\therefore$ MAX has $\alpha = 3$, MIN has $\beta = 6$

since $\beta(6) > \alpha(3)$ we stop evaluating further in this branch.

11) ∴ node with value 9 is pruned

Q.17. Explain wumpus world environment giving its PEAS description. Explain how percept sequence is generated?

→ 1) The Wumpus world is grid-based AI environment where an agent explores a cave while avoiding hazards like pits and the wumpus monster.

2) PEAS descriptor:

1) Performance Measure: Reaching the gold safely, minimizing steps.

2) Environment: A 4x4 grid with pits, gold and wumpus.

3) Actuators: move forward, turn, grab, shoot, climb.
4) sensors: perceive stench(wumpus nearby), breeze (pit nearby), glitter (gold found).
5) percept sequence generation:
1) Agent starts at (1,1) , sensing its surroundings.
2) if stench is detected, the wumpus is nearby.
3) if breeze is detected, a pit is nearby.
4) The agent infers safe paths and navigates toward the gold while avoiding hazards.

Q.18. solve the following crypto-arithmetic problem SEND + MORE = MONEY.

SEND
+ MORE
MONEY

1) Here the input size of 4 bit and output is of 5 bit. therefore, there must be carry
∴ M = 1

2) assume S = 9

∴ S + M = MO ⇒ 9 + 1 = 10 therefor we get O = 0

3) then E + O = N but O = 0 (zero)
∴ E = N but it is not possible, therefore, there must be a carry present over there.

4) ∴ E + 1 = N the N + R = E

5) assume N = 8 R = 7 ∴ N + R → 8 + 7 · 15 with 1 carry
but it is not possible
therefore assume N = 7 R = 6 N + R = 7 + 6 = 13 ∴ E = 3
but E + 1 = 3 + 1 = 4 ≠ N

6) so again assume N = 6 R = 8 ∴ N + R ⇒ 6 + 8 = 14
∴ E = 4 + 1 = 5 (add a carry)

7) Now check N + R = 8 + 7 = 15 ∴ E = 5 N = 6 R = 8

FOR EDUCATIONAL USE

0 1 2 3 4 5 6 7 8 9

8) ∴ D +E / y — Here we have value of E & S

Now assume D = 7 ∴ D + E = 7 + 5 = 12

∴ y = 2

∴ S = 9, E = 5, Y = 2, D = 12, N = 6, M = 1, O = 0, R = 8,

Q.19. consider the following axioms:

~~1) Represent these axioms in first order predicate logic (FOPL):~~

All people who are graduating are happy.

All people who are happy are smiling.

Someone is graduating.

explain the following:

1) Represent these axioms in first order predicate logic.

1) $\forall x$ (Grandmother $(x) \rightarrow$ Happy (x))

2). $\forall x$ (Happy $(x) \longrightarrow$ Smiling (x))

3) $\exists x$ (Graduating (x))

2) convert each formula to clause form.

1) $\sim$ Graduating $(x) \vee$ Happy (x)

2) $\sim$ Happy(x) $\vee$ Smiling (x)

3) Graduation (A) ← assuming A is the individual who is graduating

3) prove that "is someone smiling?" using simulation (resolution technique.

From (3) : Graduating (A)

using 1: Happy (A)

using 2: Smiling (A)

thus, someone is smiling.

4) Resolution tree can be given as:

1) Graduating (A) given

2) Graduating (A) → Happy (A), → Happy (A)

FOR EDUCATIONAL USE

3) Happy (A) → smiling (A) – smiling (A)

Q20. Explain Modus Ponens with suitable example.

→ 1) Modus Ponens (law of detachment) is a rule of interference stating:

$P \to Q, P \Rightarrow Q$.

2) eg. 1) if it rains, the ground gets wet. (premises: Rain → wet ground)

2) it is raining. (premise: Rain)

3) conclusion: the ground is wet.

Q.21. Explain forward chaining and backward chaining algorithm with an example.

→ 1) forward chaining: 1) Data-driven interference: starts from known facts and applies rules to reach a goal.

2) used in expert system (eg. medical diagnosis)

3) eg. 1) fact: "sore throat".

2) Rule: "If sore throat → infection"

3) New fact: "Infection"

4) Rule "If infection → need antibiotics".

5) conclusion: "Need antibiotics".

2) backward chaining: 1) goal-driven inference: starts from goal and works backward to find supporting facts.

2) used in AI reasoning and theorem proving

3) eg. goal: Does the patient need antibiotics?

1) check: Does the patient have an infection?

2) check: Does the patient have a sore throat?

3) If both hold, conclude "need antibiotics".

This reduces unnecessary computations by only exploring relevant facts.