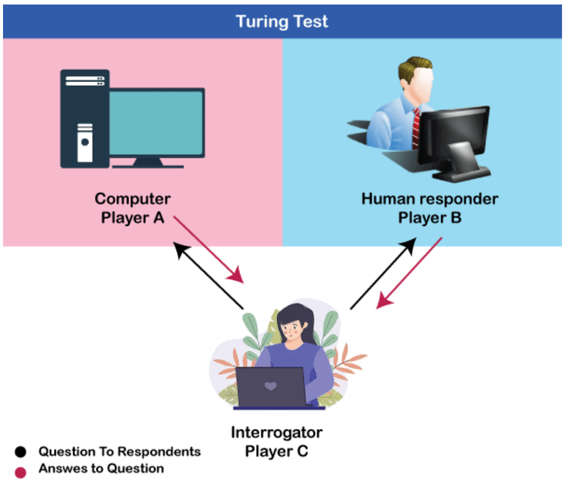
UNIT I INTRODUCTION

Introduction to Artificial Intelligence – Artificial Intelligence Problems – Timelines of Artificial Intelligence – Production Systems – State space Representation – Branches of Artificial Intelligence – Application of Artificial Intelligence.

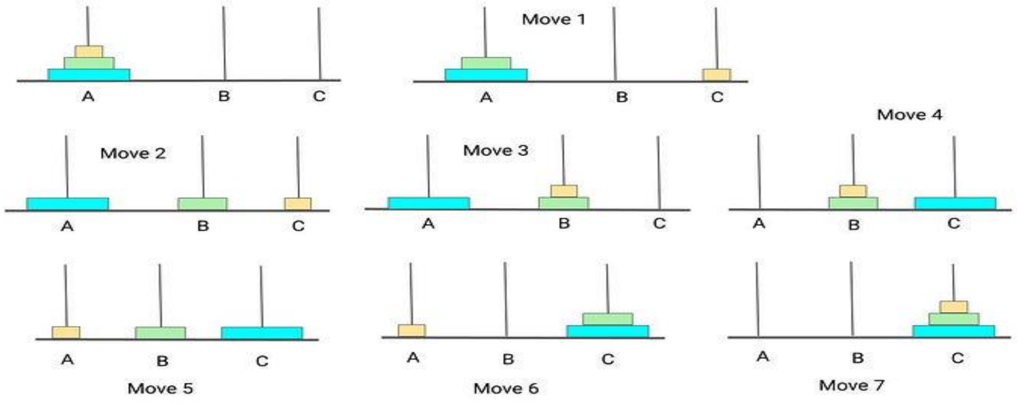
PART A

S.NO	QUESTIONS	BTL
1	**Define Artificial Intelligence.** It is a branch of computer science by which we can create intelligent machines which can behave like a human, think like humans, and able to make decisions.	K2
2	**Differentiate Human Intelligence and Artificial Intelligence.**	K2

S. No.	Feature	Artificial Intelligence	Human Intelligence
1.	Emergence	AI is an advancement made by human insights; its early improvement is credited to Norbert Weiner who theorized on criticism mechanisms.	On the other hand, human creatures are made with the intrinsic capacity to think, reason, review, etc.
2.	Nature	Artificial intelligence (AI) strives to build machines that can mimic human behavior and carry out human-like tasks.	Human intelligence seeks to adapt to new situations by combining a variety of cognitive processes.
3.	State	Machines are digital.	The human brain is analogous.
4.	Function	AI-powered machines rely on input of data and instructions.	Humans use their brains' memory, processing power, and cognitive abilities.
5.	Pace/Rate of AI and human	As compared to people, computers can handle more data at a speedier rate. For occurrence, in the event that the human intellect can solve a math problem in 5 minutes, AI can solve 10 problems in a minute.	In terms of speed, humans cannot beat the speed of AI or machines.

3	**List the types of Artificial Intelligence?** They are 3 Types of Artificial Intelligence • Artificial Narrow Intelligence (ANI) • Artificial General Intelligence (AGI) • Artificial Super Intelligence (ASI)	K2
4	**How the Turing Test can be applied to evaluate an AI system's ability to mimic human intelligence?**  The Turing test was designed to prove the satisfactory operational definition of intelligence. Consider, Player A is a computer, Player B is human, and Player C is an interrogator. Interrogator is aware that one of them is machine, but he needs to identify this on the basis of questions and their responses. The conversation between all players is via keyboard and screen so the result would not depend on the machine's ability to convert words as speech. The test result does not depend on each correct answer, but only how closely its responses like a human answer. The computer is permitted to do everything possible to force a wrong identification by the interrogator. If an interrogator would not be able to identify which is a machine and which is human, then the computer passes the test successfully, and the machine is said to be intelligent and can think like a human.	K3
5	**What is meant by Production System?** Production system or production rule system is a computer program typically used to provide some form of artificial intelligence, which consists primarily of a set of rules about behavior but it also includes the mechanism necessary to follow those rules as the system responds to states of the world.	K2
6	**Name the major branches of Artificial Intelligence?** • Machine learning • Neural Network • Robotics • Expert Systems • Fuzzy Logic	K2

	• Natural Language Processing	
7	**List some applications of Artificial Intelligence.** Google's AI-powered predictions (E.g.: Google Maps) • Ride-sharing applications (E.g.: Uber, Lyft) • AI Autopilot in Commercial Flights • Spam filters on E-mails • Plagiarism checkers and tools • Facial Recognition • Search recommendations • Voice-to-text features • Smart personal assistants (E.g.: Siri, Alexa) • Fraud protection and prevention.	K2
8	**List the components of Production System?** • Global Database • Set of Production Rules • A Control System	K2
9	**Define State Space Representation.** The production rule consists of a global database, set of rules and a goal. The initial state can be represented in the way in which computer can understand. This representation is known as **State Space Representation.** By taking up an applicable rule, we can derive another state in the solution path of the problem. The same has to be represented in the computer in a form in which it can understand. State space representation is a process in artificial intelligence (AI) that explores all possible states of a problem until it finds one with the desired feature. It's a structured way to organize and explore the problem's possible configurations, which helps AI algorithms search for solutions efficiently.	K2
10	**How to apply different machine learning techniques to a given dataset?** 1.Supervised Learning i.Classification ii.Regression 2.Unsupervised Learning i.Clustering 3.Semi-supervised/Weakly Supervised Learning 4.Reinforcement Learning	K3
11	**Apply the concept of state space to write a solution for the Tower of Hanoi problem.** • **Initial State:** All disks are on the first peg, e.g., ([1, 2, 3], [], []) for 3 disks. • **Goal State:** All disks are moved to the target peg (e.g., the third peg), e.g., ([], [], [1, 2, 3]). Given 3 disks, the output will show the sequence of states: 1. Move disk 1 to peg 3: ([2, 3], [], [1]) 2. Move disk 2 to peg 2: ([3], [2], [1]) 3. Move disk 1 to peg 2: ([3], [1, 2], []) 4. Move disk 3 to peg 3: ([], [1, 2], [3]) 5. Move disk 1 to peg 1: ([1], [2], [3]) 6. Move disk 2 to peg 3: ([1], [], [2, 3]) 7. Move disk 1 to peg 3: ([], [], [1, 2, 3])	K3



12

How to solve the Missionaries and Cannibals problem as a state space problem and outline the steps to solve it.

Each state can be represented by a tuple (M, C, B) where:

- M is the number of missionaries on the starting side (left bank).
- C is the number of cannibals on the starting side (left bank).
- B indicates the position of the boat (0 for the starting side, 1 for the destination side).
- **Initial State:** All missionaries and cannibals are on the starting side, and the boat is also on that side, e.g., `(3, 3, 0)`.
- **Goal State:** All missionaries and cannibals are on the opposite side, and the boat is also on that side, e.g., `(0, 0, 1)`.

Missionaries and Cannibals Problem (M=3 C=3 B=2)

StepId	Action
1	[2] Cannibals Leave from the Left bank
2	[1] Cannibal Returns from the Right bank
3	[2] Cannibals Leave from the Left bank
4	[1] Cannibal Returns from the Right bank
5	[2] Missionaries Leave from the Left bank
6	[1] Missionary And [1] Cannibal Return from the Right bank
7	[2] Missionaries Leave from the Left bank
8	[1] Cannibal Returns from the Right bank
9	[2] Cannibals Leave from the Left bank
10	[1] Cannibal Returns from the Right bank
11	[2] Cannibals Leave from the Left bank

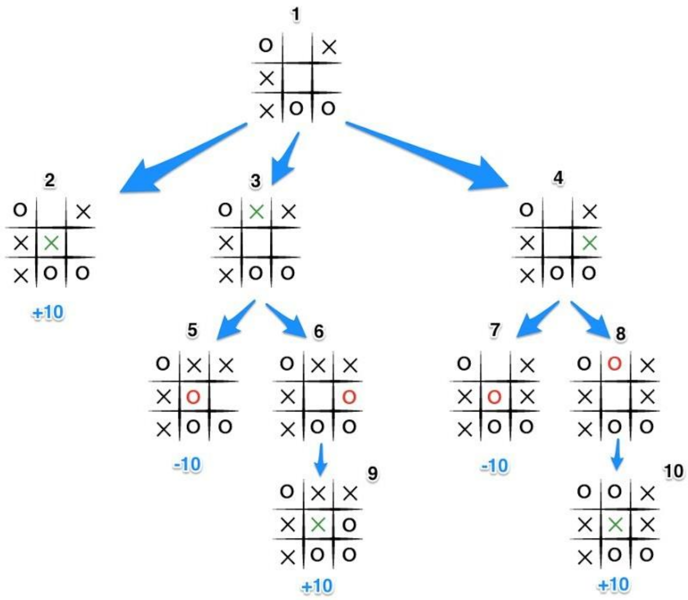
K3

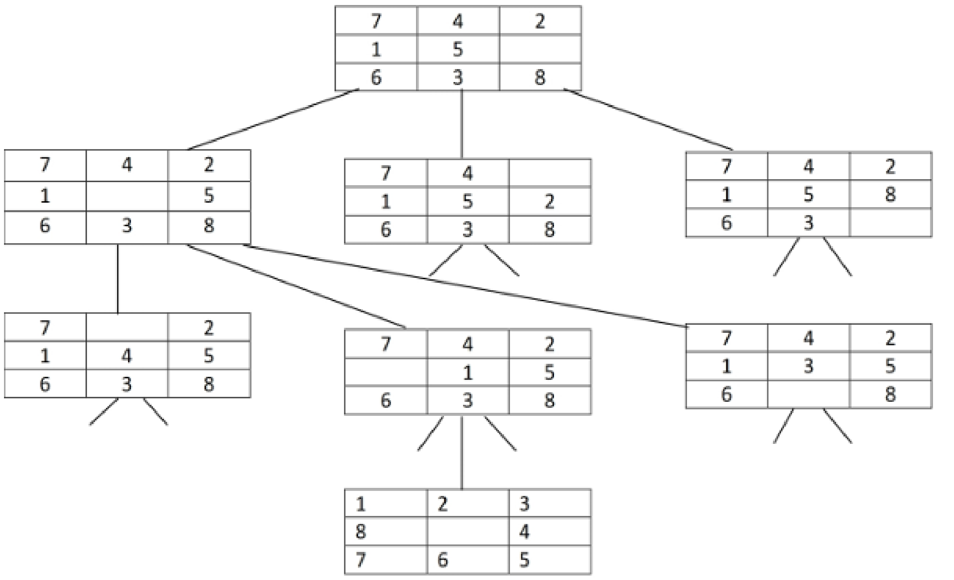
13

Illustrate Machine learning?

Machine learning is a subfield of artificial intelligence, which is broadly defined as the capability of a machine to imitate intelligent human behavior. Artificial intelligence systems are used to perform complex tasks in a way that is similar to how humans solve problems.

K3

<table>
<tr><td>14</td><td>Analyze the key characteristics that distinguish Artificial Intelligence problems from other problem types.
AI systems should be capable of learning from data or experiences and adapting their behaviour accordingly. This enables them to improve performance over time and handle new situations more effectively. Complexity: AI problems often involve dealing with complex systems or large amounts of data.</td><td>K4</td></tr>
<tr><td>15</td><td>Analyze the Tic-Tac-Toe problem as a state space representation.
Each state in Tic-Tac-Toe can be represented as a 3x3 grid, where each cell can have one of three possible values:
• X: Indicates a move by Player X.
• O: Indicates a move by Player O.
• Empty: Indicates an unoccupied cell.
The initial state is an empty 3x3 board, where no moves have been made
The goal states in Tic-Tac-Toe are the terminal states where the game ends:
• Winning States: Any configuration where three of the same symbols (X or O) align horizontally, vertically, or diagonally
Draw State: A configuration where all cells are filled, but neither player has three in a row.
Non-Terminal States: Any state where the game is still ongoing, meaning there are empty cells, and no player has won yet.

</td><td>K4</td></tr>
<tr><td colspan="3">PART B</td></tr>
<tr><td>S.NO</td><td>QUESTIONS</td><td>BTL</td></tr>
<tr><td>1</td><td>Explain gaming tree concept for 8 tile puzzle problem.
• A Game Tree is a structure for organizing all possible (legal) game states by the moves which allow transition from one game state to the next.
• This structure helps the computer to evaluate which moves to make because, by traversing the game tree, a computer (program) can easily see the outcome of a move and can decide whether to take it or not.</td><td>K2</td></tr>
</table>

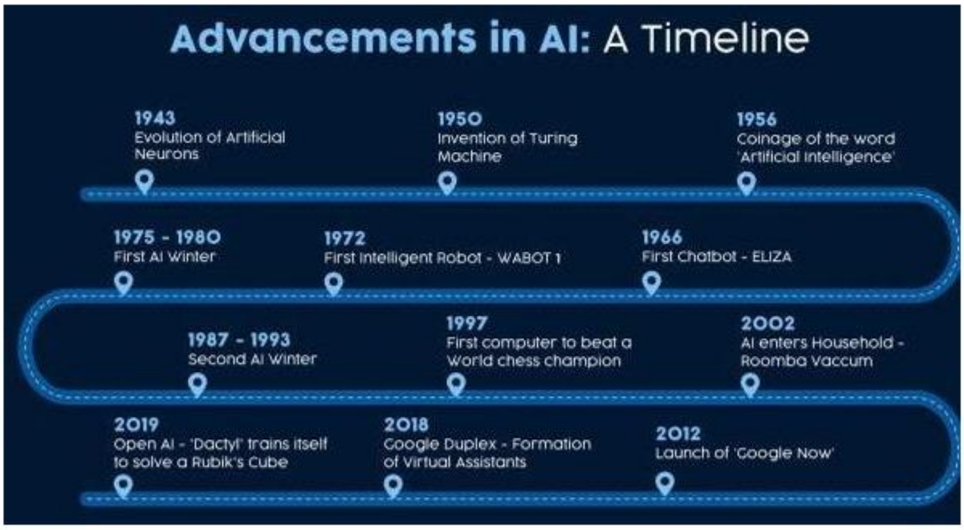
<table>
<tr>
<td></td>
<td>

The following states are used to represent a game tree 1. The board state: This is an initial stage. 2. The current player: It refers to the player who will be making the next move. 3. The next available moves: For humans, a move involves placing a game token while the computer selects the next game state. 4. The game state: It includes the grouping of the three previous concepts. 5. Final Game States: In final game states, AI should select the winning move in such a way that each move assigns a numerical value based on its board state.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Describe the artificial intelligence problem.

• To understand AI, we can define some problems that we encounter in our daily life. • Almost all the problem stated in AI commonly uses the term STATE.
• It defines the state of the solution for given problem at that particular step.
• In short, the solution of a problem by a collection of the problem state.
• The problem solving procedure used is to apply an operator to a state to get the next state.
• The process of deriving a new state from the current state by applying the operator till desired state is reached is called State space approach Problems and how it is differ from other
• If a problem need symbolic representation in computer • If there is combinational explosion in out putting
• 8 queen problem • Travelling sales man problem
• Fuzzy set for un characterize data
• The Knowledge base of an AI problem is Voluminous.
• The data or Knowledge base is Changing fast
• Doing work without tiredness and fatigue. Characteristics of AI – How the Problem is analyzed
• Is the problem decomposable or not
• Can the solution steps be ignored
• Is the solution is universe predictable
• Is the solution to a problem is absolute or relative
• Is the knowledge base consistent or not</td>
<td>K2</td>
</tr>
</table>

<table>
<tr><td></td><td>• The role of the knowledge
• Is the interaction with computer is Necessary.</td><td></td></tr>
<tr><td>3</td><td>

Explain how AI has evolved over time and why its development is timely in today's context.



</td><td>K2</td></tr>
<tr><td>4</td><td>

Given two jugs of capacity 5l and 3l (litre), and a tap with an endless supply of water. The objective is to obtain 4 litres exactly in the 5-liter jug with the minimum steps possible. Write the production rules for obtaining the solution and draw the state space representation for the problem.

1. State Representation
Each state can be represented by a tuple (x, y) where:

- x is the amount of water in the 5-liter jug.
- y is the amount of water in the 3-liter jug.

2. Initial State

- The initial state is (0, 0), where both jugs are empty.

3. Goal State

- The goal state is (4, y) where x = 4 and y can be any value.

4. Production Rules
The production rules are the possible actions that can be taken at each step:

1. **Fill the 5-liter jug**: If x < 5, fill the 5-liter jug:
   (x, y) -> (5, y)
2. **Fill the 3-liter jug**: If y < 3, fill the 3-liter jug:
   (x, y) -> (x, 3)
3. **Empty the 5-liter jug**: If x > 0, empty the 5-liter jug:
   (x, y) -> (0, y)
4. **Empty the 3-liter jug**: If y > 0, empty the 3-liter jug:
   (x, y) -> (x, 0)
5. **Pour water from the 5-liter jug into the 3-liter jug**:
   - If x > 0 and y < 3, pour water from the 5-liter jug into the 3-liter jug until either the 5-liter jug is empty or the 3-liter jug is full:
     - If x + y <= 3: (x, y) -> (0, x + y)
     - If x + y > 3: (x, y) -> (x - (3 - y), 3)
6. **Pour water from the 3-liter jug into the 5-liter jug**:
   - If y > 0 and x < 5, pour water from the 3-liter jug into the 5-liter jug until either the 3-liter jug is empty or the 5-liter jug is full:
     - If x + y <= 5: (x, y) -> (x + y, 0)
     - If x + y > 5: (x, y) -> (5, y - (5 - x))

</td><td>K3</td></tr>
</table>

5. State Space Representation

The state space representation is a graph where nodes represent states (x, y) and edges represent the production rules (actions) that transition from one state to another. The solution is a path from the initial state (0, 0) to the goal state (4, y).

6. Solution Steps

One possible sequence of moves to achieve the goal with the minimum number of steps is as follows:

1. **(0, 0) → Fill 5-liter jug → (5, 0)**
2. **(5, 0) → Pour from 5-liter jug into 3-liter jug → (2, 3)**
3. **(2, 3) → Empty 3-liter jug → (2, 0)**
4. **(2, 0) → Pour from 5-liter jug into 3-liter jug → (0, 2)**
5. **(0, 2) → Fill 5-liter jug → (5, 2)**
6. **(5, 2) → Pour from 5-liter jug into 3-liter jug → (4, 3)**
7. **(4, 3) → Goal reached (4, y)**

5. State Space Representation

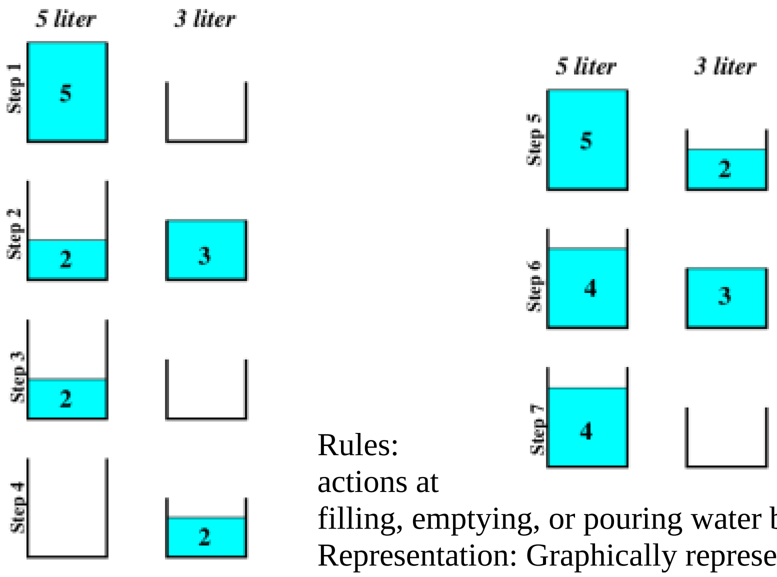
The state space representation is a graph where nodes represent states (x, y) and edges represent the production rules (actions) that transition from one state to another. The solution is a path from the initial state (0, 0) to the goal state (4, y).

6. Solution Steps

One possible sequence of moves to achieve the goal with the minimum number of steps is as follows:

1. **(0, 0) → Fill 5-liter jug → (5, 0)**
2. **(5, 0) → Pour from 5-liter jug into 3-liter jug → (2, 3)**
3. **(2, 3) → Empty 3-liter jug → (2, 0)**
4. **(2, 0) → Pour from 5-liter jug into 3-liter jug → (0, 2)**
5. **(0, 2) → Fill 5-liter jug → (5, 2)**
6. **(5, 2) → Pour from 5-liter jug into 3-liter jug → (4, 3)**
7. **(4, 3) → Goal reached (4, y)**

7. State Space Diagram



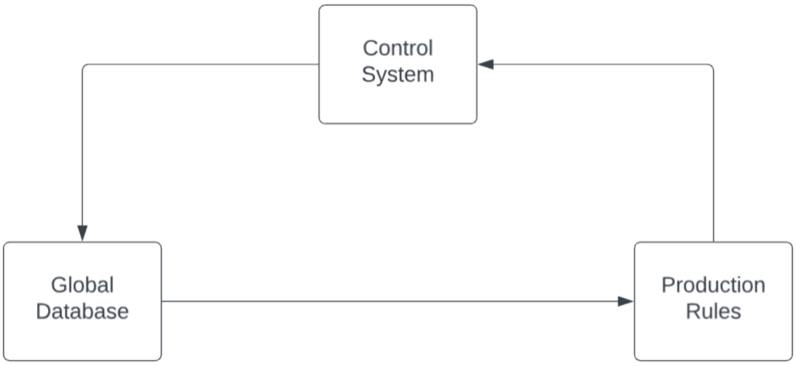
Summary

Production Rules: Describe the possible actions at each state, such as filling, emptying, or pouring water between jugs.

State Space Representation: Graphically represents the states and the transitions between them.

Minimum Steps Solution: A sequence of actions leading from the initial state (0, 0) to the goal state (4, y) in the fewest steps, illustrated by the sequence of states.

5	**Analyze how the different components and characteristics of a production system interact and contribute to its overall functionality.**	K4



The components of Production System in AI encompass three essential elements:

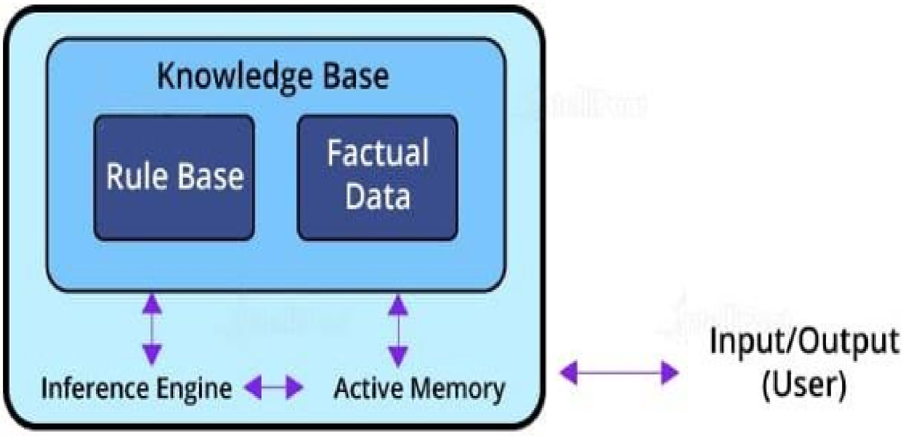
1. **Global Database / Working Memory**: Also called the global database, this is a temporary storage area that holds facts about the current state of the problem or situation being analyzed by the system.
2. **Production Rules / Knowledge Base**: This is a collection of rules that encode domain-specific knowledge. Rules typically take the form of "IF (condition) THEN (action)". For example, an expert system for medical diagnosis might have a rule like "IF the patient has a fever AND a rash, THEN there is a possibility of measles."
3. **Control System / Inference Engine**: This is the control mechanism that iteratively evaluates the rules from the knowledge base against the contents of the working memory. It determines which rules are applicable and fires (executes) them, updating the working memory with new facts derived from applying the rules.

Characteristics of Production System in AI

AI Production Systems exhibit several key features that make them versatile and powerful tools for automated decision-making and problem-solving:

- **Simplicity:** Production Systems offer a straightforward way to encode and execute rules, making them accessible for developers and domain experts.
- **Modularity:** These systems are composed of modular components, allowing for the addition, removal, or modification of rules without disrupting the entire system. This modularity enhances flexibility and ease of maintenance.
- **Modifiability:** AI Production Systems are highly adaptable. Rules can be updated or replaced without extensive reengineering, ensuring the system remains up-to-date and aligned with evolving requirements.
- **Knowledge-intensive:** They excel in handling knowledge-rich tasks, relying on a comprehensive global database.
- **Adaptability:** AI Production Systems can dynamically adapt to new data and scenarios. This adaptability allows them to continuously improve.

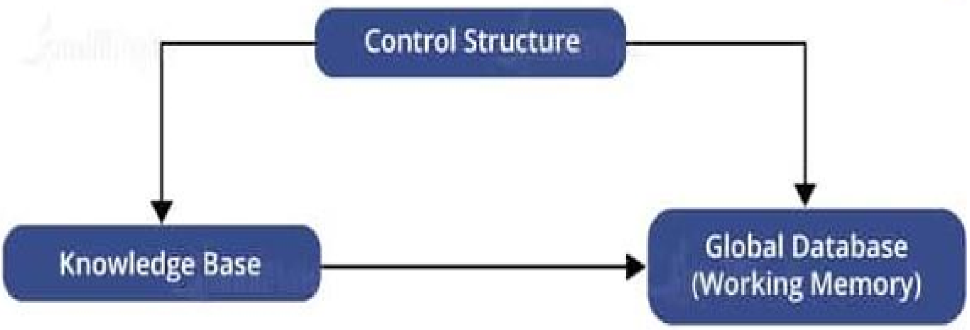
PART C

S.NO	QUESTIONS	BTL
1	**Explain the concept of a Production System in AI and its main components. Give on example problem.** A production system in AI helps create AI-based computer programs. With the help of it, the automation of various types of machines has become an easy task. The types of machines can be a computer, mobile applications, manufacturing tools, or more. The set of rules in a production system in Artificial Intelligence defines the behavior of the machine. It helps the machine respond to the surroundings. A production system in AI is a type of cognitive architecture that defines specific actions as per certain rules. The rules represent the declarative knowledge of a machine to respond according to different conditions. Today, many expert systems and automation methodologies rely on the rules of production systems. Below is the basic architecture of production systems in AI: 	K2

The rules in a production system are determined by LHS (left-hand side) and RHS (right-hand side) equations, where LHS denotes the specific condition to be applied, and RHS shows the output of the applied condition.

Components of a Production System in AI

For making an AI-based intelligent system that performs specific tasks, we need an architecture. The architecture of a production system in Artificial Intelligence consists of production rules, a database, and the control system.



Let us discuss each one of them in detail.

Global Database

A global database consists of the architecture used as a central data structure. A database contains all the necessary data and information required for the successful completion of a task. It can be divided into two parts as permanent and temporary. The permanent part of the database consists of fixed actions, whereas the temporary part alters according to circumstances.

Production Rules

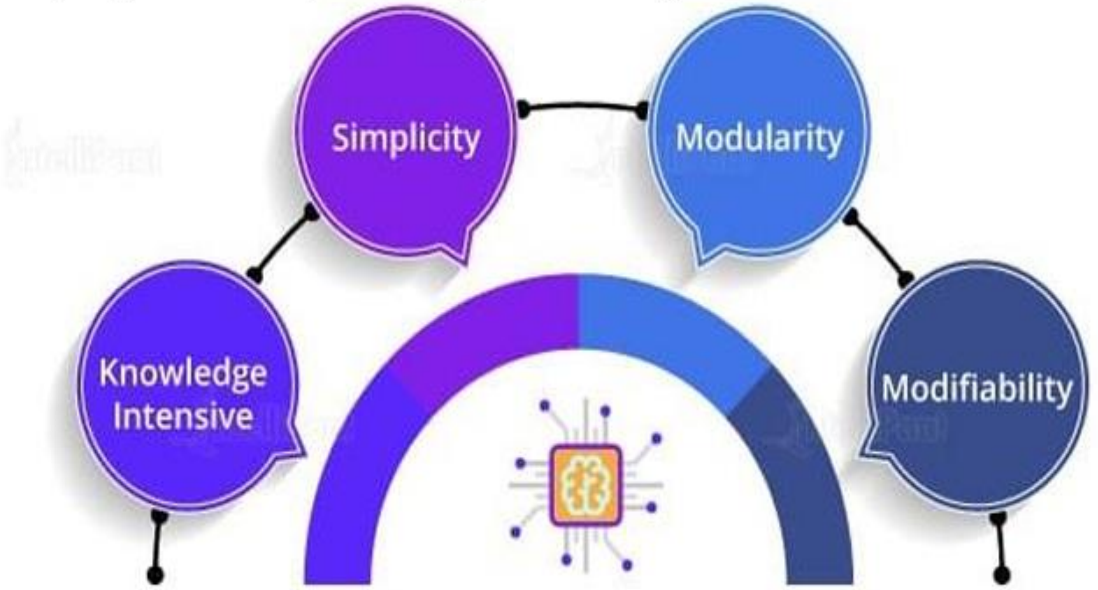
Production rules in AI are the set of rules that operates on the data fetched from the global database. Also, these production rules are bound with precondition and postcondition that gets checked by the database. If a condition is passed through a production rule and gets satisfied by the global database, then the rule is successfully applied. The rules are of the form A®B, where the right-hand side represents an outcome corresponding to the problem state represented by the left-hand side.

Control System

The control system checks the applicability of a rule. It helps decide which rule should be applied and terminates the process when the system gives the correct output. It also resolves the conflict of multiple conditions arriving at the same time. The strategy of the control system specifies the sequence of rules that compares the condition from the global database to reach the correct result.

Characteristics of a Production System

There are mainly four characteristics of the production system in AI that is simplicity, modifiability, modularity, and knowledge-intensive.



Simplicity

The production rule in AI is in the form of an 'IF-THEN' statement. Every rule in the production system has a unique structure. It helps represent knowledge and reasoning in the simplest way possible to solve real-world problems. Also, it helps improve the readability and understanding of the production rules.

Modularity

The modularity of a production rule helps in its incremental improvement as the production rule can be in discrete parts. The production rule is made from a collection of information and facts that may not have dependencies unless there is a rule connecting them together. The addition or deletion of single information will not have a major effect on the output. Modularity helps enhance the performance of the production system by adjusting the parameters of the rules.

Modifiability

The feature of modifiability helps alter the rules as per requirements. Initially, the skeletal form of the production system is created. We then gather the requirements and make changes in the raw structure of the production system. This helps in the iterative improvement of the production system.

Knowledge-intensive

Production systems contain knowledge in the form of a human spoken language, i.e., English. It is not built using any programming languages. The knowledge is represented in plain English sentences. Production rules help make productive conclusions from these sentences.

Production System in Artificial Intelligence: Example

Problem Statement:

- We have two jugs of capacity 5l and 3l (liter), and a tap with an endless supply of water. The objective is to obtain 4 liters exactly in the 5-liter jug with the minimum steps possible

Production System:

1. Fill the 5 liter jug from tap
2. Empty the 5 liter jug
3. Fill the 3 liter jug from tap
4. Empty the 3 liter jug
5. Then, empty the 3 liter jug to 5 liter
6. Empty the 5 liter jug to 3 liter
7. Pour water from 3 liters to 5 liter
8. Pour water from 5 liters to 3 liters but do not empty

Solution:

- **1,8,4,6,1,8** or **3,5,3,7,2,5,3,5;**

Advantages

- Some of the **advantages** of Production system in artificial intelligence are:
- Provides **excellent tools** for structuring AI programs
- The system is highly **modular** because individual rules can be added, removed or modified independently
- Separation of **knowledge** and **Control-Recognises Act Cycle**
- A natural **mapping** onto state-space research data or goal-driven
- The system uses pattern directed control which is more **flexible** than algorithmic control
- Provides opportunities for **heuristic control** of the search
- A good way to model the **state-driven nature** of intelligent machines
- Quite helpful in **a real-time** environment and applications

Disadvantages

- It is very **difficult** to analyze the flow of control within a production system
- It describes the operations that can be performed in a search for a solution to the problem.
- There is an **absence of learning** due to a rule-based production system that does not store the result of the problem for future use.
- The rules in the production system should not have any type of **conflict resolution** as when a new rule is added to the database it should ensure that it does not have any conflict with any existing rule.

2	**A man carries a cabbage, a goat and a wolf. He wants to across a river and there is a boat available, but he can take only one item at a time. In absence of man, wolf may eat goat or goat may eat cabbage. How can he take the entire three to the other side of the river. Solve this problem using production system.**	K4

1. State Representation

Each state can be represented as a tuple (M, C, G, W) where:

- M represents the position of the man (left or right bank).
- C represents the position of the cabbage.
- G represents the position of the goat.
- W represents the position of the wolf.

Each of these elements can be in either of two positions:

- L (Left bank)
- R (Right bank)

2. Initial State

- The initial state is (L, L, L, L) where all items (man, cabbage, goat, and wolf) are on the left bank.

3. Goal State

- The goal state is (R, R, R, R) where all items are safely on the right bank.

4. Production Rules

The production rules define valid moves the man can make with or without an item. They must ensure that after each move, the goat is not left alone with the cabbage, and the wolf is not left alone with the goat.

Valid Moves:

1. **Man crosses alone:** (M, C, G, W) -> (M', C, G, W)
2. **Man takes cabbage:** (M, M, G, W) -> (M', M', G, W)
3. **Man takes goat:** (M, C, M, W) -> (M', C, M', W)
4. **Man takes wolf:** (M, C, G, M) -> (M', C, G, M')

Constraints:

- The goat cannot be left with the cabbage (i.e., (M, L, L, W) or (M, R, R, W) is invalid).
- The wolf cannot be left with the goat (i.e., (M, C, L, L) or (M, C, R, R) is invalid).

5. Control Strategy

The control strategy involves systematically applying the rules to move from the initial state to the goal state, while avoiding invalid states.

6. Solution Steps

Here's the step-by-step solution using the production system:

1. **(L, L, L, L) → (R, L, R, L)**
   - **Man takes the goat** across the river to the right bank.
2. **(R, L, R, L) → (L, L, L, L)**
   - **Man returns alone** to the left bank.
3. **(L, L, L, L) → (R, R, L, L)**
   - **Man takes the cabbage** across the river to the right bank.
4. **(R, R, L, L) → (L, R, R, L)**
   - **Man brings the goat back** to the left bank.
5. **(L, R, R, L) → (R, R, R, R)**
   - **Man takes the wolf** across the river to the right bank.
6. **(R, R, R, R) → (L, R, L, R)**
   - **Man returns alone** to the left bank.
7. **(L, R, L, R) → (R, R, R, R)**
   - **Man takes the goat** across the river to the right bank.

Final Sequence:

1. **(L, L, L, L) → (R, L, R, L)** (Man takes goat)
2. **(R, L, R, L) → (L, L, L, L)** (Man returns alone)

<table>
<tr>
<td></td>
<td>

3. **(L, L, L, L) → (R, R, L, L)** (Man takes cabbage)
4. **(R, R, L, L) → (L, R, R, L)** (Man brings goat back)
5. **(L, R, R, L) → (R, R, R, R)** (Man takes wolf)
6. **(R, R, R, R) → (L, R, L, R)** (Man returns alone)
7. **(L, R, L, R) → (R, R, R, R)** (Man takes goat)

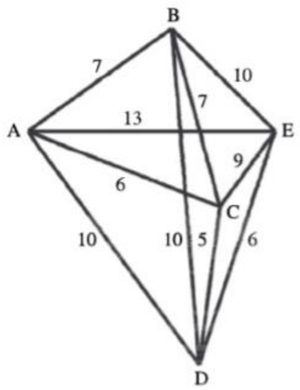
7. State Space Representation

The state space can be visualized as a graph where nodes represent states (M, C, G, W) and edges represent valid moves:

```
(L, L, L, L) -> (R, L, R, L) -> (L, L, L, L) -> (R, R, L, L)
                      |                               |
                      v                               v
            (L, R, R, L) -> (R, R, R, L) -> (L, R, L, R)
                                                      |
                                                      v
                                              (R, R, R, R)
```

</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>

Enumerate the global database for the following problem and find a solution. A salesman must visit each of the 5 cities as shown in below figure. There is a road between every pair of cities and the corresponding distance are given. The problem is to find a minimal path that he visits each of the cities only one and returns to the starting city.



</td>
<td>K3</td>
</tr>
</table>

Enumerate the global database for the following problem and find a solution.

A salesman must visit each of the five cities as shown in Figure 1.4. There is a road between every pair of cities and the corresponding distances are given. The problem is to find a minimal path that he visits each of the cities only once and returns to the starting city.

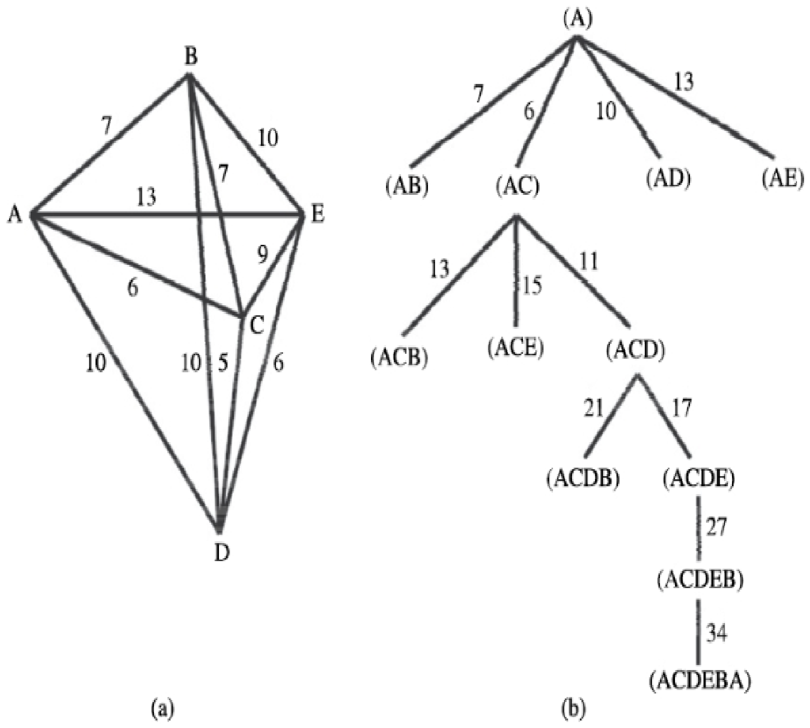


Figure 1.4 (a) 5 cities connected; (b) global database.

The goal is to find the minimum path. The rule is to branch out from a node which has minimal distance from the root node.

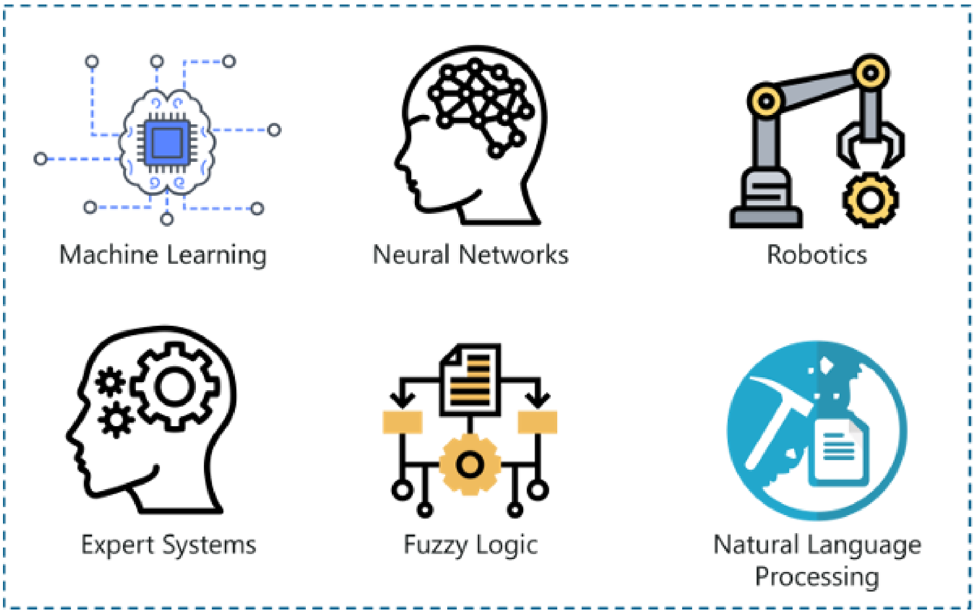
Solution

Let A be selected as the starting node. We can specify the global database in the form of a graph. The minimum path is A C D E B A and Cost = 34. Figure 1.4(b) shows the global database for the travelling salesman problem.

4	**Explain and briefly tell about the Branches of Artificial Intelligence?**	K2

- Machine learning
- Neural Network
- Robotics
- Expert Systems
- Fuzzy Logic
- Natural Language Processing

There is a broad set of techniques that come in the domain of artificial intelligence such as linguistics, bias, vision, planning, robotic process automation, natural language processing, decision science, etc. Let us acquire information about some of the major subfields of AI in deep;



Machine learning

- Machine Learning is the technique that gives computers the potential to learn without being programmed, it is actively being used in daily life, machine learning applications in daily life, even without knowing that. Fundamentally, it is the science that enables machines to translate, execute and investigate data for solving real-world problems.

Types of machine learning

- **Supervised Learning:** In this type of learning, data experts feed labelled training data to algorithms and define variables to algorithms for accessing and finding correlations. Both the input and output of the algorithm are particularized/defined.
- **Unsupervised Learning:** This type of learning include algorithms that train on unlabelled data, an algorithm analyzes datasets to draw meaningful correlations or inferences. For example, one method is cluster analysis that uses exploratory data analysis to obtain hidden or grouping patterns or groups in datasets.
- **Reinforcement Learning:** For teaching a computer machine to fulfil a multi-step process for which there are clearly defined rules, reinforcement learning is practised. Here, programmers design an algorithm to perform a task and give it positive and negative signal to act as algorithm execute to complete the task. Sometimes, the algorithm even determines on its own what action to take to go ahead.

Neural Network

- In simple terms, a neural network is a set of algorithms that are used to find the elemental relationships across the bunches of data via the process that imitates the human brain operating process.

- Neural network replicates the human brain where the human brain comprises an infinite number of neurons and to code brain-neurons into a system or a machine is what the neural network functions

Robotics

- Robotics is an interdisciplinary field of science and engineering incorporated with mechanical engineering, electrical engineering, computer science, and many others.
- Robotics determines the designing, producing, operating, and usage of robots. It deals with computer systems for their control, intelligent outcomes, and information transformation

Expert Systems

- An expert system refers to a computer system that mimics the decision-making intelligence of a human expert. It conducts this by deriving knowledge from its knowledge base by implementing reasoning and insights rules in terms with the user queries.
- The effectiveness of the expert system completely relies on the expert's knowledge accumulated in a knowledge base. The more the information collected in it, the more the system enhances its efficiency. For example, the expert system provides suggestions for spelling and errors in Google Search Engine

Fuzzy Logic

- In the real world, sometimes we face a condition where it is difficult to recognize whether the condition is true or not, their fuzzy logic gives relevant flexibility for reasoning that leads to inaccuracies and uncertainties of any condition.

	• It is simply the generalization of the standard logic where a concept exhibits a degree of truth between 0.0 to 1.0. If the concept is completely true, standard logic is 1.0 and 0.0 for the completely false concept. But in fuzzy logic, there is also an intermediate value too which is partially true and partially false. **Natural Language Processing** • NLP is the part of computer science and AI that can help in communicating between computer and human by natural language. It is a technique of computational processing of human languages. It enables a computer to read and understand data by mimicking human natural language • NLP is a method that deals in searching, analyzing, understanding and deriving information from the text form of data. In order to teach computers how to extract meaningful information from the text data, NLP libraries are used by programmers. A common example of NLP is spam detection, computer algorithms can check whether an email is a junk or not by looking at the subject of a line, or text of an email.	
5	**Explain about the Application of AI in various fields in today's world?** Artificial Intelligence has various applications in today's society. It is becoming essential for today's time because it can solve complex problems with an efficient way in multiple industries, such as Healthcare, entertainment, finance, education, etc. AI is making our daily life more comfortable and fast. Following are some sectors which have the application of Artificial Intelligence: **1. AI in Astronomy** ○ Artificial Intelligence can be very useful to solve complex universe problems. AI technology can be helpful for understanding the universe such as how it works, origin, etc. **2. AI in Healthcare** ○ In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry. ○ Healthcare Industries are applying AI to make a better and faster diagnosis than humans. AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization. **3. AI in Gaming** ○ AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.	K2

4. AI in Finance

- AI and finance industries are the best matches for each other. The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

5. AI in Data Security

- The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world. AI can be used to make your data more safe and secure. Some examples such as AEG bot, AI2 Platform,are used to determine software bug and cyber-attacks in a better way.

6. AI in Social Media

- Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way. AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

7. AI in Travel & Transport

- AI is becoming highly demanding for travel industries. AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers. Travel industries are using AI-powered chatbots which can make human-like interaction with customers for better and fast response.

8. AI in Automotive Industry

- Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.
- Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

9. AI in Robotics

- Artificial Intelligence has a remarkable role in Robotics. Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI, we can create intelligent robots which can perform tasks with their own experiences without pre-programmed.
- Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

	10. AI in Entertainment ○ We are currently using some AI based applications in our daily life with some entertainment services such as Netflix or Amazon. With the help of ML/AI algorithms, these services show the recommendations for programs or shows. **11. AI in Agriculture** ○ Agriculture is an area which requires various resources, labor, money, and time for best result. Now a day's agriculture is becoming digital, and AI is emerging in this field. Agriculture is applying AI as agriculture robotics, solid and crop monitoring, predictive analysis. AI in agriculture can be very helpful for farmers. **12. AI in E-commerce** ○ AI is providing a competitive edge to the e-commerce industry, and it is becoming more demanding in the e-commerce business. AI is helping shoppers to discover associated products with recommended size, color, or even brand. **13. AI in education** ○ AI can automate grading so that the tutor can have more time to teach. AI chatbot can communicate with students as a teaching assistant. ○ AI in the future can be work as a personal virtual tutor for students, which will be accessible easily at any time and any place.	

UNIT II KNOWLEDGE REPRESENTATION

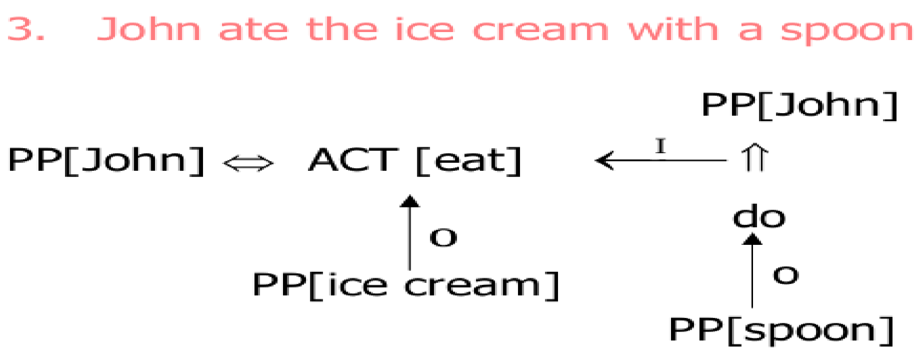
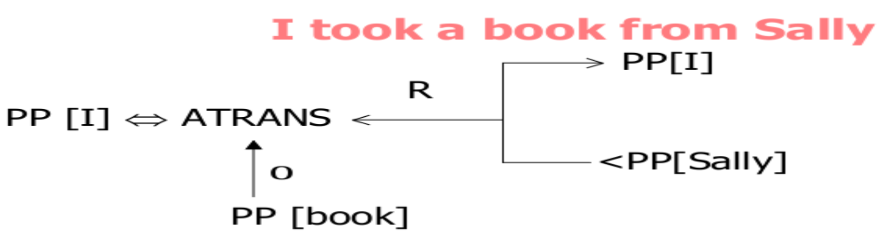
Knowledge Management - Types of Knowledge - Knowledge representation - Approaches to Knowledge representation - Issues in Knowledge representation - Knowledge base. First order Logic – Frames - Conceptual Dependency.

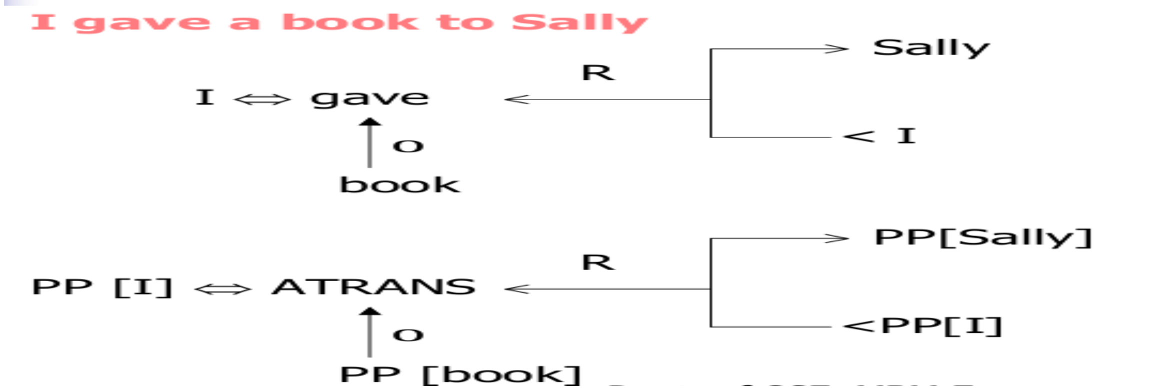
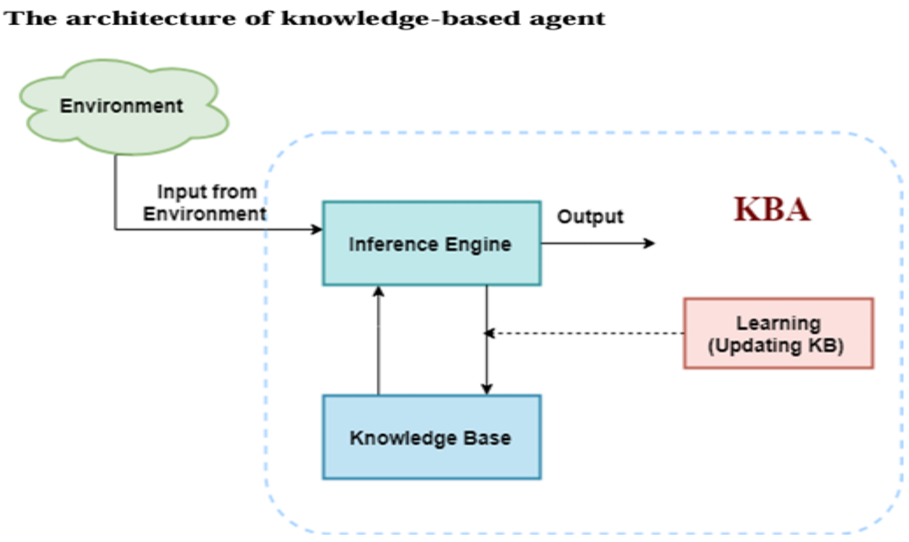
PART A

S.NO	QUESTIONS	BTL
1	**Define Knowledge Management.** The knowledge management (KM) process comprises a set of activities for identification, gathering, creation, presentation and distribution of knowledge for the purposes of learning, reuse, and awareness.	K1
2	**List the types of Knowledge.** • Declarative Knowledge • Procedural Knowledge • Meta-knowledge • Heuristic knowledge • Structural knowledge	K1
3	**Define Knowledge Representation.**	K1

	• Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behaviour of agents. • It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.	
4	**List out the Issues in Knowledge Representation.** 1. Important Attributes 2. Relationship among attributes 3. Properties of Attributes 4. Choosing Granularity 5. Set of objects 6. Finding Right structure	K1
5	**Differentiate declaration Knowledge and Procedural Knowledge.** **Procedural Knowledge:** • It is also known as imperative knowledge. • Procedural knowledge is a type of knowledge which is responsible for knowing how to do something. • It can be directly applied to any task. • It includes rules, strategies, procedures, agendas, etc. • Procedural knowledge depends on the task on which it can be applied. **Example:** It's basically "how" you know to do something. The classic example of procedural knowledge is riding a bicycle. When someone was teaching you how to ride a bicycle, no matter what they said, you probably struggled to grasp it until you'd actually done it a few times. **Declarative knowledge:** • Declarative knowledge is to know about something. • It includes concepts, facts, and objects. • It is also called descriptive knowledge and expressed in declarative sentences. • It is simpler than procedural language.	K2
6	**State the semantic Network Representation.** • Semantic networks are alternative of predicate logic for knowledge representation. In Semantic networks, we can represent our knowledge in the form of graphical networks.	K2

	• This network consists of nodes representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects. • Semantic networks are easy to understand and can be easily extended.	
7	**Illustrate Frames with example.** • A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world. Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations. • It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called facets.	K3
8	**Name the major parts of First Order Logic.** **Basic Elements of First-order logic** Constant 1, 2, A, John, Mumbai, cat,.... • Variables x, y, z, a, b,.... • Predicates Brother, Father, >,.... • Function sqrt, LeftLegOf, ... • Connectives ∧, ∨, ¬, ⇒, ⇔ • Equality == • Quantifier ∀, ∃	K1
9	**Solve the given statement using First Order Logic** **Not all Students like both Maths and Science.** ∃x(Student(x)∧(¬LikesMaths(x)∨¬LikesScience(x)))	K3
10	**Modify the given statement using into FOL** **Some integers are even and some are odd.** ∃x(Integer(x)∧Even(x))∧∃y(Integer(y)∧Odd(y))	K3
11	**Differentiate Universal and Existential Quantifier with examples.** **Universal quantifier:** It is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing. The Universal quantifier is represented by a symbol ∀, which resembles an inverted A. If x is a variable, then ∀x is read as:	K2

	• For all x • For each x • For every x. **Existential quantifier:** they are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something. It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier. If x is a variable, then existential quantifier will be ∃x or ∃(x • There exists a 'x.' • For some 'x.' • For at least one 'x.'	
12	**Solve the following statements using Conceptual Dependency** **i) John took the book.** **ii) John ate the ice cream with a spoon.**  	K3
13	**Apply Conceptual Dependency representation for the following** **i) I took a book from Sally.** **ii) I gave a book to Sally.** 	K3

<table>
<tr><td></td><td>
</td><td></td></tr>
<tr><td>14</td><td>Draw the architecture of Knowledge base agent.

</td><td>K4</td></tr>
<tr><td>15</td><td>Classify the ways of Knowledge Representation.
• Logical Representation
• Semantic Network Representation
• Frame Representation
• Production Rules</td><td>K4</td></tr>
<tr><td colspan="3">PART B</td></tr>
<tr><td>S.NO</td><td>QUESTIONS</td><td>BTL</td></tr>
<tr><td>1</td><td>Explain the different types of Knowledge Representation.

</td><td>K2</td></tr>
</table>

Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.
- It is also called descriptive knowledge and expressed in declarativesentences.
- It is simpler than procedural language.
- An example of a declarative knowledge statement is, "A car has four tires." Declarative knowledge is explicit, meaning either a person knows it or they don't.

Procedural Knowledge

- It is also known as imperative knowledge.
- Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- It can be directly applied to any task.
- It includes rules, strategies, procedures, agendas, etc.
- Procedural knowledge depends on the task on which it can be applied.
- The classic example of procedural knowledge is riding a bicycle. When someone was teaching you how to ride a bicycle, no matter what they said, you probably struggled to grasp it until you'd actually done it a few times.

Meta-knowledge:

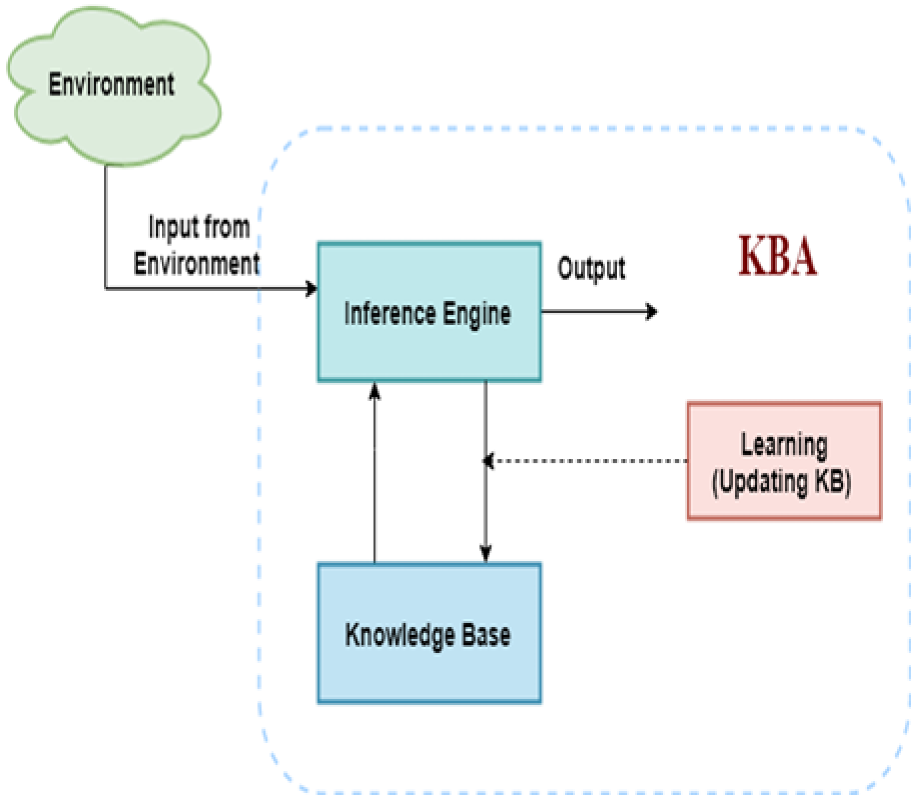
- Knowledge about the other types of knowledge is called Meta-knowledge.
- It is knowledge about knowledge, and it's important for knowledge representation, information retrieval, and improving AI system efficiency. It involves understanding how different pieces of information relate to each other, and it can help judge the validity or appropriateness of knowledge.

Structural knowledge:

- Structural knowledge is basic knowledge to problem-solving.
- It describes relationships between various concepts such as kind of, part of, and grouping of something.
- It describes the relationship that exists between concepts or objects.
- Examples include temporal relations, spatial relations, family relations, social relations, administrative organizations, military hierarchies, etc.

	Heuristic knowledge: • Heuristic knowledge is representing knowledge of some experts in a filed or subject. • Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed. • Some examples of heuristic knowledge are a hypothesis, common sense, rule of thumb, and intuition. Heuristic knowledge helps a person make judgments in a sufficient manner and amount of time. A concrete example of heuristic knowledge would be when a plumber comes to give an estimate to a new customer.	
2	**Describe the Knowledge Base Agent in Artificial Intelligence.** Knowledge-based agents are those agents who have the capability of maintaining an internal state of knowledge, reason over that knowledge, update their knowledge after observations and take actions. These agents can represent the world with some formal representation and act intelligently. • An intelligent agent needs knowledge about the real world for taking decisions and reasoning to act efficiently. • Knowledge-based agents are those agents who have the capability of maintaining an internal state of knowledge, reason over that knowledge, update their knowledge after observations and take actions. These agents can represent the world with some formal representation and act intelligently • Knowledge-based agents are composed of two main parts: a. Knowledge-base and b. Inference system. • A knowledge-based agent must able to do the following: • An agent should be able to represent states, actions, etc. • An agent Should be able to incorporate new percepts • An agent can update the internal representation of the world • An agent can deduce the internal representation of the world • An agent can deduce appropriate actions	K2

The architecture of knowledge-based agent



Inference system

- Inference means deriving new sentences from old. Inference system allows us to add a new sentence to the knowledge base. A sentence is a proposition about the world. Inference system applies logical rules to the KB to deduce new information.
- Inference system generates new facts so that an agent can update the KB. An inference system works mainly in two rules which are given as:

  a. Forward chaining
  b. Backward chaining

Operations Performed by KBA

- Following are three operations which are performed by KBA in order to show the intelligent behavior:

	a. TELL: This operation tells the knowledge base what it perceives from the environment. b. ASK: This operation asks the knowledge base what action it should perform. c. Perform: It performs the selected action **Various levels of knowledge-based agent** 1. Knowledge level 2. Logical level 3. Implementation level	
3	**Explain in detail about the 2 types of Quantifiers with examples.** **Quantifiers in First-order logic** 1. A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse. 2. These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. 3. There are two types of quantifier: • Universal Quantifier, (for all, everyone, everything) • Existential quantifier, (for some, at least one). **Universal quantifier:** It is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing. The Universal quantifier is represented by a symbol ∀, which resembles an inverted A. If x is a variable, then ∀x is read as: • For all x • For each x • For every x. **Examples** 1.All birds fly. In this question the predicate is "fly(bird)." And since there are all birds who fly so it will be represented as follows. $\forall x\ bird(x) \rightarrow fly(x)$. 2.Every man respects his parent. In this question, the predicate is "respect(x, y)," where x=man, and y= parent. Since there is every man so will use ∀, and it will be represented as follows: $\forall x\ man(x) \rightarrow respects\ (x, parent)$.	K2

	Existential quantifier: they are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something. It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier. If x is a variable, then existential quantifier will be ∃x or ∃(x • There exists a 'x.' • For some 'x.' • For at least one 'x.' **Examples:** **1. Some boys play cricket.** The predicate is "play(x, y)," where x= boys, and y= game. Since there are some boys so we will use ∃, and it will be represented as: **∃x boys(x) → play(x, cricket).** **2. Only one student failed in Mathematics.** The predicate is "failed(x, y)," where x= student, and y= subject. Since there is only one student who failed in Mathematics, so we will use following representation for this: **∃(x) [student(x) → failed (x, Mathematics) ∧∀ (y) [¬(x==y) ∧ student(y) → ¬failed (x, Mathematics)].**	
4	**Modify the given statement into First Order Logic** **i) All penguins are birds.** **ii) All Information Technology students are smart** **iii) Not every Gardner likes the sun.** **iv) John is not tall.** **v) Some birds lives in Antartica.** **i) All penguins are birds.** **FOL Statement:** ∀x(Penguin(x)→Bird(x)) • **Explanation:** For all individuals xxx, if xxx is a penguin, then xxx is also a bird. **ii) All Information Technology students are smart.** **FOL Statement:** ∀x(ITStudent(x)→Smart(x)) • **Explanation:** For all individuals xxx, if xxx is an Information Technology student, then xxx is smart. **iii) Not every gardener likes the sun.** **FOL Statement:** ¬∀x(Gardener(x)→LikesSun(x)) or ∃x(Gardener(x)∧¬LikesSun(x)) • **Explanation:** There exists at least one individual xxx such that xxx is a gardener and xxx does not like the sun. **iv) John is not tall.** **FOL Statement:** ¬Tall(John) • **Explanation:** John is not tall.	K3

	v) Some birds live in Antarctica. **FOL Statement:** $\exists x(Bird(x) \wedge LivesInAntarctica(x))$ • **Explanation:** There exists at least one individual xxx such that xxx is a bird and xxx lives in Antarctica	
5	**Analyze the railway ticket reservation and write the script of railway ticket reservation.** A script is a structured representation describing a stereotyped sequence of events in a particular context. It was originally designed by Schank in 1977. The knowledge base representation is in terms of the situations that the system is supposed to understand. They are extremely beneficial because • The events always tend to occur in known patterns or runs. • There is always a relationship between the existing events. • There exists an entry condition(s) which allows the event to take place. • There are prerequisites for the events that are taking place. For example, when a student progresses through a degree scheme or when a customer buys a product. The components of scripts are as follows: • **Entry conditions** include open conditions, satisfied before the events are described. • **Result** or 'fact' is true once the script has terminated. • **Props** or the 'things' support the content of a script. The set of props supports reasonable default assumptions about a situation. • **Roles** are actions that an individual participant performs. • **Tracks** are the specific variation on a more general pattern that is represented by a particular script. • **Scenes** are the actual sequence of events that occur. The events described in a script form a giant causal chain. A set of entry conditions are enabled in the beginning of the chain which enables the first event of the script to occur. The end of the chain is the set of results which may enable later events or event sequences to occur. Within the chain, the events are connected to earlier events that make them possible and to later events that they enable.	K4

Example: A railway ticket reservation script

Script : Railway station

Track : Ticket booking

Props : Counters

D = Display board with information

F = Ticket

Credit card

Money

Roles:

S = Traveller

W = Person in the ticket counter (Current tickets)

C = Person issues Tatkal reservation tickets

M = Ticket examiner

O = Railway management

Every condition:

S required a ticket

S has money

Result:

S has less money

O has more money

S not need reservation

S is pleased (optional)

Scene 1: Entering

S PTRANS S into station

S ATTEND eyes to counter

S MBUILD where to take

S PTRANS S to counter

S MOVE S to ticket counter

Scene 2: Reserving

(D on his hand) (W point out the D) (S asked for D)

S PTRANS Menu to S

↓

S MTRANS signal to W

W PTRANS W to counter

S MTRANS need D to W

W PTRANS W to D

↙ W PTRANS W to counter

*S MBUILD choice of F

S MTRANS signal to W

W PTRANS W to counter

S MTRANS 'I want F' to W

W MTRANS 'Tatkal is available' to S

S MTRANS 'I want Tatkal F' to W

W PTRANS W to C

W MTRANS (A TRANS F) to C

↙ ↘

C MTRANS 'Print F' to W

W PRANS W to S

W MTRANS 'Print F' to S

(go back to *)

C DO (prepare F script)

to scene 3

Scene 3 : Checking

C ATRANS F to W

W ATRANS F to S

S INGEST F

(Option: returns to scene 2 to more tickets, otherwise go to scene 4)

↙ ↘

Scene 4: Leaving

(W ATRANS money to S)

W PTRANS W to S

W ATRANS credit card to S

S PTRANS S to M

S ATRANS money to M

S PTRANS S move to out of the station

S MTRANS to W

W Move (give credit card)

Some additional points to note on scripts:

1. For applying a script, it need to be activated, and the activation depends on th significance of the particular script.
2. If a subject is noted in passing, then a pointer to that script could be held.
3. For important topics, the script should be opened.
4. It is not desirable in many active scripts having too many windows open on the displa or too many recursive calls in a program.
5. For events following known trail we can use active scripts to represent the action involved and use them to answer detailed questions.
6. Different trails may be allowed for different outcomes of scripts (e.g., the restauran script).

Advantages

- Ability to predict events.
- A single coherent interpretation may be build up from a collection of observations.

Disadvantages

- It is not as general as frames.
- It is not suitable to represent all kinds of knowledge.

PART C		
S.NO	**QUESTIONS**	**BTL**
1	**Explain the issues in Knowledge Representation.** **Issues that should be raised when using a knowledge representation technique:** • Are any attributes of objects so basic that they occur in almost every problem domain? • Are there any important relationships that exist among attributes of objects? • At what level should knowledge be represented? Is there a good set of primitives into which all knowledge can be broken down? • How should sets of objects be represented? • Given a large amount of knowledge stored in a database, how can relevant parts be accessed when they are needed? **Important Attributes:** • Any attribute of objects so basic that they occur in almost every problem domain ? a. Relationship among attributes: • Any important relationship that exists among object attributes ? a. Choosing Granularity : • At what level of detail should the knowledge be represented ? a. Set of objects : • How sets of objects be represented? a. Finding Right structure : **1. Important Attributes:** Any attribute of objects so basic that they occur in almost every problem domain? There are two attributed "instance" and "isa", that are general significance. These attributes are important because they support property inheritance **2. Relationship among attributes:** Any important relationship that exists among object attributed? • The attributes we use to describe objects are themselves entities that we represent. • The relationship between the attributes of an object, independent of specific knowledge they encode, may hold properties like:	K2

3. Properties of Attributes

1. Inverse — This is about consistency check, while a value is added to one attribute. The entities are related to each other in many different ways.

2. Existence in an isa hierarchy — This is about generalization-specification, like, classes of objects and specialized subsets of those classes, For example, the attribute height is a specialization of general attribute physical-size which is, in turn, a specialization of physical-attribute. They support inheritance

3. Technique for reasoning about values — This is about reasoning values of attributes not given explicitly. Several kinds of information are used in reasoning, like, height: must be in a unit of length, Age: of a person cannot be greater than the age of person's parents.

4. Single valued attributes — This is about a specific attribute that is guaranteed to take a unique value. For example, a baseball player can at time have only a single height and be a member of only one team.

4. Choosing Granularity

- At what level of detail should the knowledge be represented?
- Regardless of the KR formalism, it is necessary to know:
- At what level should the knowledge be represented and what are the primitives?
- Should there be a small number or should there be a large number of low-level primitives or High-level facts.
- High-level facts may not be adequate for inference while Low-level primitives may require a lot of storage.

Example of Granularity:

- John spotted Sue.
- This could be represented as
- Spotted (agent(John),object (Sue))
- Such a representation would make it easy to answer questions such are:
- Who spotted Sue?
- Suppose we want to know:
- Did John see Sue?

- Given only one fact, we cannot discover that answer.
- We can add other facts, such as
- Spotted(x, y) -> saw(x, y)

5. Set of objects

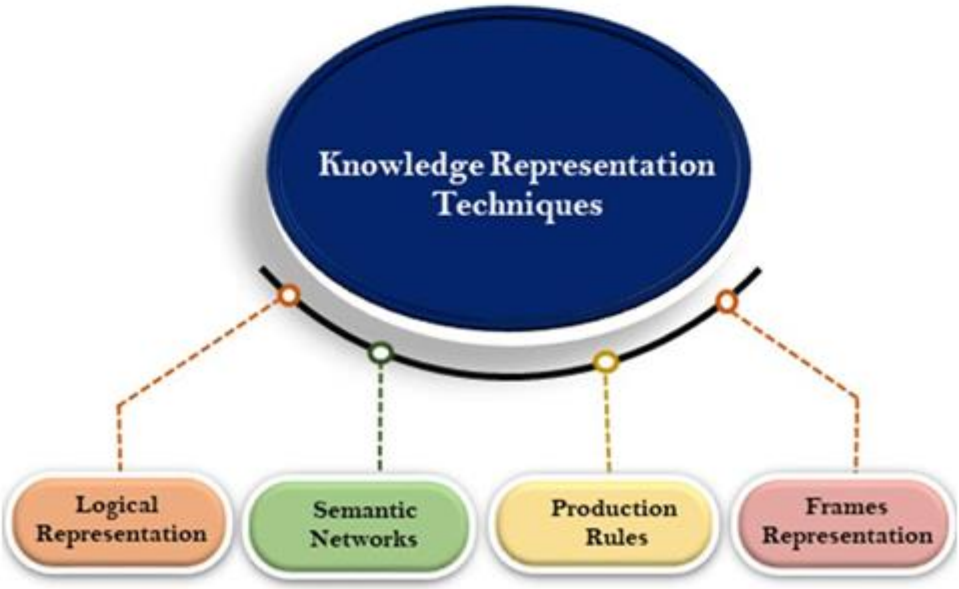
- How should sets of objects be represented?
- There are certain properties of objects that are true as member of a set but not as individual;
- Example: Consider the assertion made in the sentences:
- "there are more sheep than people in Australia", and
- "English speakers can be found all over the world."
- To describe these facts, the only way is to attach assertion to the sets representing people, sheep, and English.
- The reason to represent sets of objects is: if a property is true for all or most elements of a set, then it is more efficient to associate it once with the set rather than to associate it explicitly with every elements of the set.
- This is done,in logical representation through the use of universal quantifier, and
- in hierarchical structure where node represent sets and inheritance propagate set level assertion down to individual.

6. Finding Right structure

- Given a large amount of knowledge stored in a database, how can relevant parts are accessed when they are needed?
- This is about access to right structure for describing a particular situation.
- This requires, selecting an initial structure and then revising the choice

While doing so, it is necessary to solve following problems

- How to perform an initial selection of the most appropriate structure.
- How to fill in appropriate details from the current situations.
- How to find a better structure if the one chosen initially turns out not to be appropriate.
- What to do if none of the available structures is appropriate.
- When to create and remember a new structure.

2	**Explain the techniques in Knowledge Representation.** There are mainly four ways of knowledge representation which are given as follows: • Logical Representation • Semantic Network Representation • Frame Representation • Production Rules  	K2

1. **Logical Representation**

Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation. Logical representation means drawing a conclusion based on various conditions. This representation lays down some important communication rules. It consists of precisely defined syntax and semantics which supports the sound inference. Each sentence can be translated into logics using syntax and semantics.

Syntax:

- Syntaxes are the rules which decide how we can construct legal sentences in the logic.
- It determines which symbol we can use in knowledge representation.
- How to write those symbols.

Semantics:

- Semantics are the rules by which we can interpret the sentence in the logic.
- Semantic also involves assigning a meaning to each sentence.

Logical representation can be categorised into mainly two logics:

- Propositional Logics
- Predicate logics

Advantages of logical representation:

1. Logical representation enables us to do logical reasoning.
2. Logical representation is the basis for the programming languages.

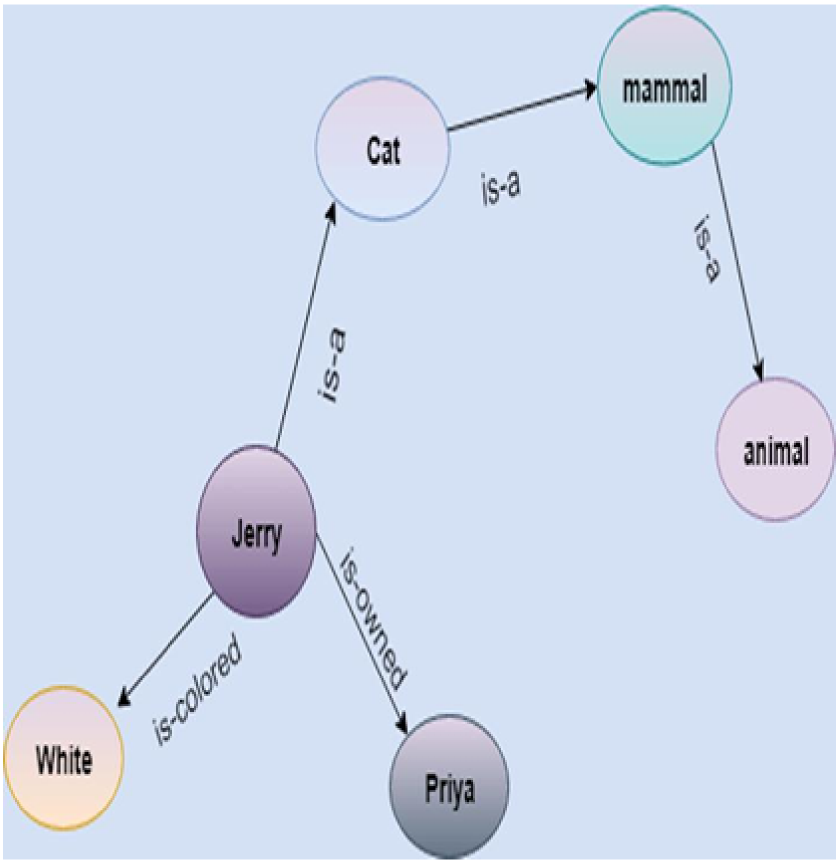
Disadvantages of logical Representation:

1. Logical representations have some restrictions and are challenging to work with.
2. Logical representation technique may not be very natural, and inference may not be so efficient.

3. **Semantic Network Representation**

Semantic networks are alternative of predicate logic for knowledge representation. In Semantic networks, we can represent our knowledge in the form of graphical networks. This network consists of nodes

	representing objects and arcs which describe the relationship between those objects. Semantic networks can categorize the object in different forms and can also link those objects. Semantic networks are easy to understand and can be easily extended. This representation consists of mainly two types of relations: a. IS-A relation (Inheritance) b. Kind-of-relation Example: Following are some statements which we need to represent in the form of nodes and arcs. Statements: a. Jerry is a cat. b. Jerry is a mammal c. Jerry is owned by Priya. d. Jerry is brown colored. e. All Mammals are animal.	



In the above diagram, we have represented the different type of knowledge in the form of nodes and arcs. Each object is connected with another object by some relation.

Drawbacks in Semantic representation:

1. Semantic networks take more computational time at runtime as we need to traverse the complete network tree to answer some questions. It might be possible in the worst case scenario that after traversing the entire tree, we find that the solution does not exist in this network.
2. Semantic networks try to model human-like memory (Which has 1015 neurons and links) to store the information, but in practice, it is not possible to build such a vast semantic network.
3. These types of representations are inadequate as they do not have any equivalent quantifier, e.g., for all, for some, none, etc.
4. Semantic networks do not have any standard definition for the link names.

5. These networks are not intelligent and depend on the creator of the system.

Advantages of Semantic network:

1. Semantic networks are a natural representation of knowledge.
2. Semantic networks convey meaning in a transparent manner.
3. These networks are simple and easily understandable.

3. Frame Representation

A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world. Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations. It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called facets.

Slots	Filters
Title	Artificial Intelligence
Genre	Computer Science
Author	Peter Norvig
Edition	Third Edition

Year	1996
Page	1152

Facets: The various aspects of a slot is known as Facets. Facets are features of frames which enable us to put constraints on the frames. Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets and facets may have any number of values. A frame is also known as slot-filter knowledge representation in artificial intelligence.

Frames are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not much useful. Frames system consist of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base. The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Advantages of frame representation:

- The frame knowledge representation makes the programming easier by grouping the related data.
- The frame representation is comparably flexible and used by many applications in AI.
- It is very easy to add slots for new attribute and relations.
- It is easy to include default data and to search for missing values.
- Frame representation is easy to understand and visualize.

Disadvantages of frame representation:

- In frame system inference mechanism is not be easily processed.
- Inference mechanism cannot be smoothly proceeded by frame representation.
- Frame representation has a much generalized approach.

4. Production Rules

Production rules system consist of (condition, action) pairs which mean, "If condition then action". It has mainly three parts:

- The set of production rules
- Working Memory
- The recognize-act-cycle

In production rules agent checks for the condition and if the condition exists then production rule fires and corresponding action is carried out. The condition part of the rule determines which rule may be applied to a problem. And the action part carries out the associated problem-solving steps. This complete process is called a recognize-act cycle.

The working memory contains the description of the current state of problems-solving and rule can write knowledge to the working memory. This knowledge match and may fire other rules.

If there is a new situation (state) generates, then multiple production rules will be fired together, this is called conflict set. In this situation, the agent needs to select a rule from these sets, and it is called a conflict resolution.

Example:

- IF (at bus stop AND bus arrives) THEN action (get into the bus)
- IF (on the bus AND paid AND empty seat) THEN action (sit down).
- IF (on bus AND unpaid) THEN action (pay charges).
- IF (bus arrives at destination) THEN action (get down from the bus).

Advantages of Production rule:

a. The production rules are expressed in natural language.
b. The production rules are highly modular, so we can easily remove, add or modify an individual rule.

Disadvantages of Production rule:

a. Production rule system does not exhibit any learning capabilities, as it does not store the result of the problem for the future uses.
b. During the execution of the program, many rules may be active hence rule-based production systems are inefficient.

3	**Explain in detail about First Order Logic.**
	First-Order Logic
	• In Propositional logic, we represent statements using propositional logic. But unfortunately, in propositional logic, we can only represent the facts, which are either true or false.
	• PL is not sufficient to represent the complex sentences or natural language statements. The propositional logic has very limited expressive power. Consider the following sentence, which we cannot represent using PL logic.
	• "Some humans are intelligent", or
	"Sachin likes cricket."
	• To represent the above statements, PL logic is not sufficient, so we required some more powerful logic, such as first-order logic.
	  • First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
	  • FOL is sufficiently expressive to represent the natural language statements in a concise way.
	  • First-order logic is also known as Predicate logic or First-order predicate logic. First-order logic is a powerful language that develops information about the objects in a more easy way and can also express the relationship between those objects.
	  • First-order logic (like natural language) does not only assume that the world contains facts like propositional logic but also assumes the following things in the world:
	a. Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ...
	b. Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between
	c. Function: Father of, best friend, third inning of, end of,
	
	As a natural language, first-order logic also has two main parts:
	a. Syntax
	b. Semantics
	
	Syntax of First-Order logic:
		K2

The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Basic Elements of First-order logic

Constant 1, 2, A, John, Mumbai, cat,....

- Variables x, y, z, a, b,....
- Predicates Brother, Father, >,....
- Function sqrt, LeftLegOf, ...
- Connectives ∧, ∨, ¬, ⇒, ⇔
- Equality ==
- Quantifier ∀, ∃

Atomic sentences

Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by a parenthesis with a sequence of terms.

We can represent atomic sentences as Predicate (term1, term2, ..., term n).

Example: Ravi and Ajay are brothers:

=> Brothers(Ravi, Ajay).

Chinky is a cat:

=> cat (Chinky).

Complex Sentences

- Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
- **Subject:** Subject is the main part of the statement.
- **Predicate:** A predicate can be defined as a relation, which binds two atoms together in a statement.

- Consider the statement: "x is an integer.", it consists of two parts, the first part x is the subject of the statement and second part "is an integer," is known as a predicate.

Quantifiers in First-order logic

1. A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.

2. These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression.

3. There are two types of quantifier:

•Universal Quantifier, (for all, everyone, everything)

•Existential quantifier, (for some, at least one).

Universal Quantifier

1. Universal quantifier is a symbol of logical representation, which specifies that the statement within its range is true for everything or every instance of a particular thing.

2. The Universal quantifier is represented by a symbol $\forall$, which resembles an inverted A.

3.If x is a variable, then $\forall x$ is read as:

•For all x

•For each x

•For every x.

Examples

1. All birds fly.

In this question the predicate is "fly(bird)."

And since there are all birds who fly so it will be represented as follows.

$$\forall x \ bird(x) \rightarrow fly(x).$$

2. Every man respects his parent.

<table>
<tr><td></td><td>

In this question, the predicate is "respect(x, y)," where x=man, and y= parent.

Since there is every man so will use ∀, and it will be represented as follows:

$\forall x\ man(x) \rightarrow respects\ (x, parent)$.

Existential Quantifier

1. Existential quantifiers are the type of quantifiers, which express that the statement within its scope is true for at least one instance of something.

2. It is denoted by the logical operator ∃, which resembles as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.

3. If x is a variable, then existential quantifier will be ∃x or ∃(x)

•There exists a 'x.'

•For some 'x.'

•For at least one 'x.'

Example

1. Some boys play cricket.

The predicate is "play(x, y)," where x= boys, and y= game. Since there are some boys so we will use ∃, and it will be represented as:

$\exists x\ boys(x) \rightarrow play(x, cricket)$.

2. Only one student failed in Mathematics.

The predicate is "failed(x, y)," where x= student, and y= subject.

Since there is only one student who failed in Mathematics, so we will use following representation for this:

$\exists(x)\ [\ student(x) \rightarrow failed\ (x, Mathematics) \land \forall\ (y)\ [\neg(x{=}{=}y) \land student(y) \rightarrow \neg failed\ (x, Mathematics)]$.

</td><td></td></tr>
<tr><td>4</td><td>

Construct Conceptual Dependency representation of the following

i) John gave Mary a pencil.
ii) Reji took the book from Jim.
iii) Vinod stirred his Coffee with a Spoon.
iv) Ram killed Ravan.
v) Sacin did not go to restaurant as terrorists attacked Mumbai.

</td><td>K3</td></tr>
</table>

John gave Mary a pencil

$$\underset{PP}{\text{John}} \overset{P}{\Longleftrightarrow} \text{ATRANS} \overset{O}{\longleftarrow} \underset{PP}{\text{Pencil}} \overset{R}{\longleftarrow} \begin{cases} \rightarrow \text{Mary (PP)} \\ \leftarrow \text{John (PP)} \end{cases}$$

Reji took the book from Jim.

$$\underset{PP}{\text{Reji}} \overset{P}{\Longleftrightarrow} \text{ATRANS} \overset{O}{\longleftarrow} \underset{PP}{\text{book}} \overset{R}{\longleftarrow} \begin{cases} \rightarrow \text{Reji} \\ \leftarrow \text{Jim} \end{cases}$$

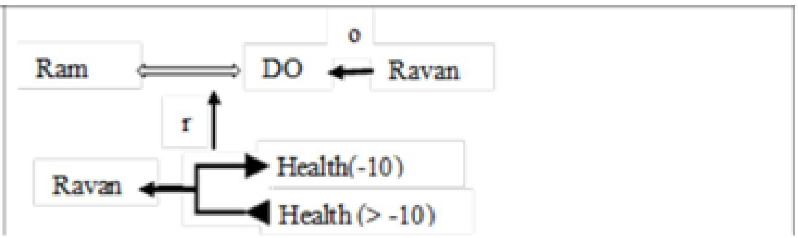
Vinod stirred his coffee with a spoon.

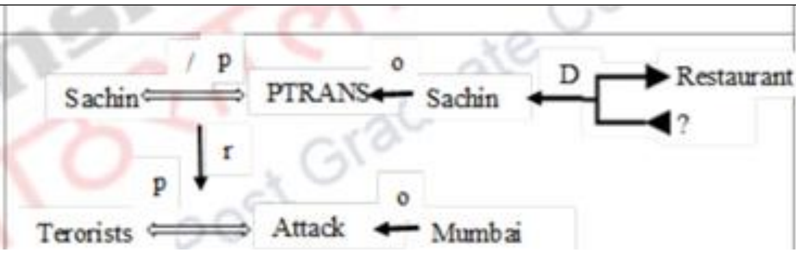
$$\underset{PP}{\text{Vinod}} \overset{P}{\Longleftrightarrow} \text{PROPEL} \overset{O}{\longleftarrow} \underset{PP}{\text{Coffee}}$$

$$\uparrow I$$

$$\underset{PP}{\text{Vinod}} \Longleftrightarrow \text{do} \overset{O}{\longleftarrow} \underset{PP}{\text{Spoon}}$$

. Ram Killed Ravan



	Sachin did not go to restaurent as terrorists attacked Mumbai 	
5	**Analyze the restaurant scenes and write a script for restaurant.** A script is a structured representation describing a stereotyped sequence of events in a particular context. It was originally designed by Schank in 1977. The knowledge base representation is in terms of the situations that the system is supposed to understand. They are extremely beneficial because · The events always tend to occur in known patterns or runs. · There is always a relationship between the existing events. · There exists an entry condition(s) which allows the event to take place. · There are prerequisites for the events that are taking place. For example, when a student progresses through a degree scheme or when a customer buys a product. The components of scripts are as follows: · **Entry conditions** include open conditions, satisfied before the events are described. · **Result** or 'fact' is true once the script has terminated. · **Props** or the 'things' support the content of a script. The set of props supports reasonable default assumptions about a situation. · **Roles** are actions that an individual participant performs. · **Tracks** are the specific variation on a more general pattern that is represented by a particular script. · **Scenes** are the actual sequence of events that occur. The events described in a script form a giant causal chain. A set of entry conditions are enabled in the beginning of the chain which enables the first event of the script to occur. The end of the chain is the set of results which may enable later events or event sequences to occur. Within the chain, the events are connected to earlier events that make them possible and to later events that they enable.	K4

Script: Restaurant
Track: Coffee Shop
Props: Tables, menus, food, check, money
Roles: Customer, cook, owner, waiter, cashier

Entry Conditions: Customer is hungry.
Customer has money.

Results: Customer has less money.
Customer is not hungry.
Owner has more money.

Scenes:

1. Entering

Customer goes into restaurant.
Customer looks around.
Customer decides where to sit.
Customer goes to the table and sits down.

2. Ordering

Customer picks up menus.
Customer decides on food.
Customer orders food from waiter.
Waiter tells cook the order.
Cook prepares food.

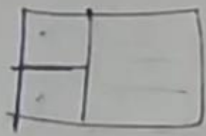
3. Eating

Cook gives food to waiter.
Waiter gives food to customer.
Customer eat food.

4. Exiting

Waiter writes out check.
Waiter brings check to customer.
Customer gives tip to waiter.
Customer goes to cash register.
Customer gives money to cashier.
Customer leaves restaurant.

Script for Resturant

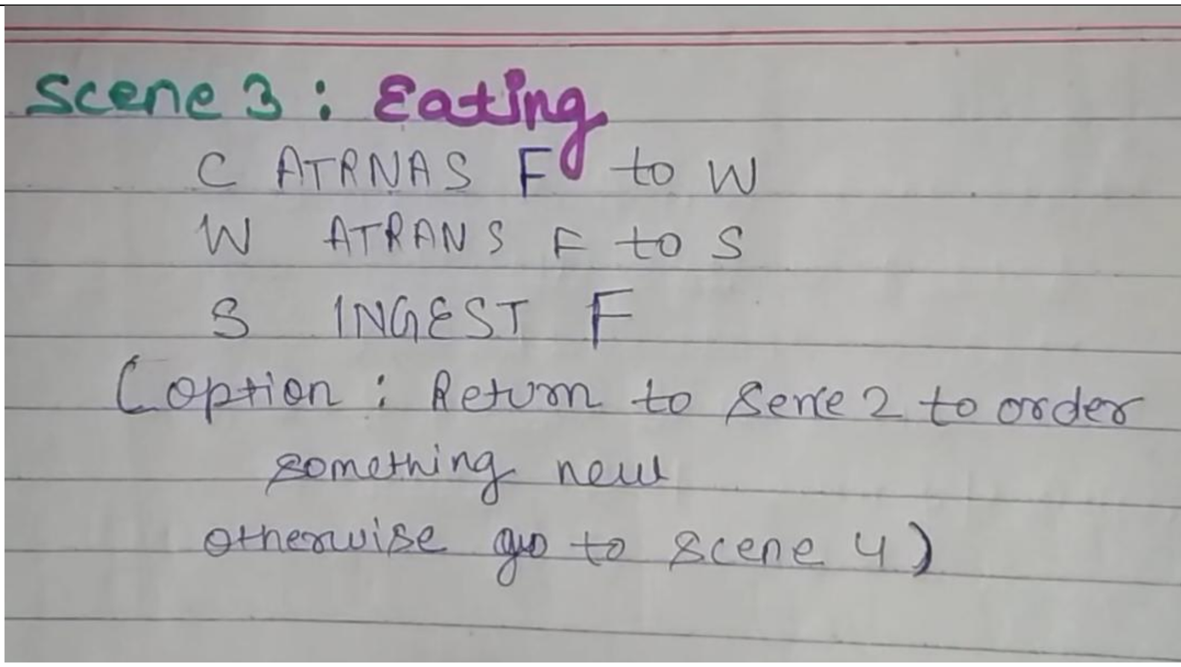


Scene 1 : Entering

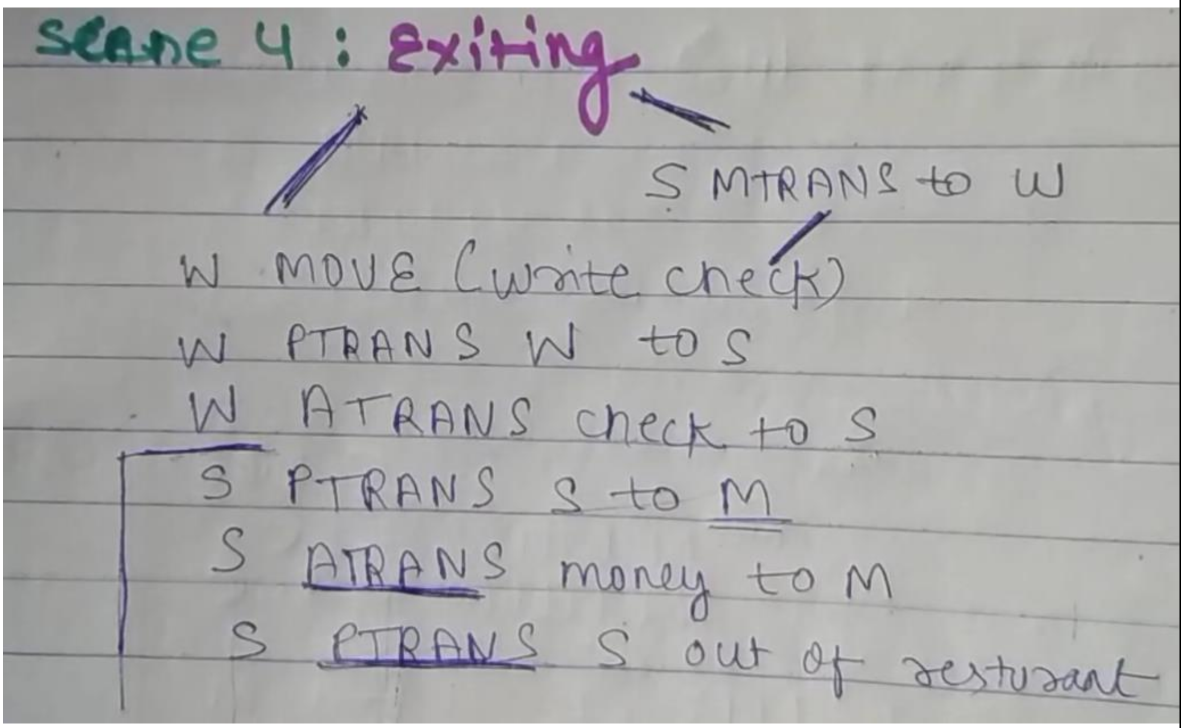
S PTRANS S into resturant
S ATTEND eyes to tables
S MBUILD where to sit
S PTRANS S to table
S MOVE S to sitting position

Scene 2 : ordering

(W brings menu) (menu on table)
S ATRANS menu to S
S MTRANS W to table
S MBUILD choice of F
S MTRANS signal to W
W PTRANS W to table
S MTRANS 'I want F' to W
W PTRANS W to C
W MTRANS to C
C DO
to Scene 3



Scene 3 : Eating
C ATRNAS F to W
W ATRANS F to S
S INGEST F
(Option : Return to Scene 2 to order something new
otherwise go to scene 4)



Scene 4 : Exiting
S MTRANS to W
W MOVE (write check)
W PTRANS W to S
W ATRANS check to S
S PTRANS S to M
S ATRANS money to M
S PTRANS S out of restuarant

UNIT III - REASONING

Types of reasoning - reasoning with Fuzzy Logic - Rule based Reasoning - Diagnosis Reasoning.

PART A

S.NO	QUESTIONS	BTL
1	**What is reasoning in the context of Artificial Intelligence?**	**K1**
	• In **AI**, reasoning refers to the process of making decisions or drawing conclusions based on data or rules. • It helps AI systems solve problems, infer new information, and make logical decisions. **Deductive reasoning** applies known rules to specific cases, while **inductive reasoning** finds general patterns from observations.	

	• **Abductive reasoning** suggests the most likely explanation for an event. Reasoning allows AI to mimic human-like thinking and decision-making.	
2	**Difference between deductive and inductive reasoning?**	**K3**

Feature	Deductive Reasoning	Inductive Reasoning
Direction of Logic	General rule → Specific case	Specific observations → General conclusion
Certainty	Conclusions are certain if premises are true	Conclusions are probable, not guaranteed
Example	"All birds have wings; a sparrow is a bird, so a sparrow has wings."	"After observing many sparrows, all have wings; therefore, all sparrows probably have wings."
Application	Used in mathematics, logic, rule-based systems	Common in science and discovering patterns
Conclusion	Definite and logically follows from premises	Probable generalization based on evidence

3	**What is the role of uncertainty in Fuzzy Logic?**	**K1**

- Uncertainty in **Fuzzy Logic** allows systems to handle vague and imprecise information, unlike traditional binary logic.
- It assigns **degrees of membership** (between 0 and 1) to elements, reflecting partial truths.
- This enables **flexible decision-making**, even when data is incomplete or uncertain.
- Fuzzy logic systems respond **smoothly** to changing inputs, making them effective in control applications.
- It also mimics **human reasoning**, which often deals with ambiguity.
- This makes Fuzzy Logic useful in real-world scenarios like **medical diagnosis** and robotics, where precise data isn't always available.
- Overall, uncertainty enhances the **adaptability and robustness** of AI systems.

4	**How does Rule-based Reasoning differ from case-based reasoning?**	**K2**

Aspect	Rule-based Reasoning	Case-based Reasoning
Foundation	Based on a predefined set of rules or logical statements.	Based on past cases or experiences similar to the current situation.
Knowledge Structure	Uses **if-then** rules derived from general knowledge.	Relies on a **database of past cases** or examples.
Approach	Applies general rules to deduce conclusions for specific problems.	Solves new problems by comparing them to previously solved cases.
Flexibility	Less flexible, depends on having the right rules for all situations.	More flexible, adapts solutions based on case similarities.
Learning	New knowledge must be encoded as new rules.	System learns and improves by storing and adapting new cases.
Example	An expert system diagnosing a medical condition using predefined rules.	A system diagnosing a patient by comparing their symptoms to past patient cases.

5	**What is the purpose of a rule engine in rule-based reasoning systems?**	K1

The purposes of a **rule engine** in rule-based reasoning systems:

1. **Rule Management**: Stores and organizes rules ("if-then" statements) for decision-making.
2. **Inference Processing**: Evaluates rules against data to draw conclusions.
3. **Decouples Logic**: Separates rule logic from application code for easier updates.
4. **Dynamic Decisions**: Allows real-time responses to changing conditions.
5. **Conflict Resolution**: Manages situations where multiple rules apply by prioritizing them.
6. **Scalability**: Efficiently handles large sets of rules and data for complex applications.
7. **Logging**: Tracks rule execution for auditing and understanding decisions.

6	**Mention some main components in rule-based system.**	K2

The main components of a rule-based system:

1. Knowledge Base
2. Rule Engine
3. Inference Engine
4. Working Memory
5. User Interface
6. Explanation Facility
7. Conflict Resolution Strategy

The database

External interface

7	**Difference between crisp logic and fuzzy logic?**	K3

Aspect	Crisp Logic	Fuzzy Logic
Definition	Uses binary values (true/false or 0/1)	Allows for degrees of truth (between 0 and 1)
Precision	Precise and deterministic	Handles uncertainty and vagueness
Membership	An element either belongs or does not belong to a set (full membership)	An element can have partial membership in a set
Reasoning	Based on classical logic rules	Based on approximate reasoning
Applications	Suitable for clear-cut problems, like mathematical calculations	Useful in complex, real-world scenarios, like control systems
Complexity	Simpler and easier to understand	More complex due to handling of imprecise information
Example	A statement is either "hot" or "not hot"	A statement can be "mostly hot" (e.g., 0.7 hot)

8	**Define an IF-THEN rule in the context of rule-based reasoning with example.**	K1

An **IF-THEN rule** in the context of rule-based reasoning is a conditional statement that specifies an action or conclusion based on a certain condition or set of conditions. The structure consists of two parts:

- **IF part (Antecedent)**: This specifies the condition(s) that need to be met.
- **THEN part (Consequent)**: This specifies the action or conclusion that follows if the conditions in the IF part are satisfied.

Example of an IF-THEN Rule

Rule:

- **IF** the temperature is greater than 30°C **THEN** turn on the air conditioner.

Explanation

- **IF part**: "the temperature is greater than 30°C" (condition)
- **THEN part**: "turn on the air conditioner" (action)

9	**What does the term degree of membership mean in fuzzy logic?**	**K1**

In fuzzy logic, the **degree of membership** refers to the extent to which an element belongs to a fuzzy set. Unlike traditional binary sets, where an element either belongs to a set or does not (with values of 0 or 1), fuzzy sets allow for values between 0 and 1, representing partial membership.

Range: The degree of membership is a value between 0 and 1.

- A value of **0** means the element does not belong to the set at all.
- A value of **1** means the element fully belongs to the set.
- Values between 0 and 1 indicate varying degrees of belonging.

10	**Difference between forward chaining and backward chaining in rule-based reasoning?**	**K3**

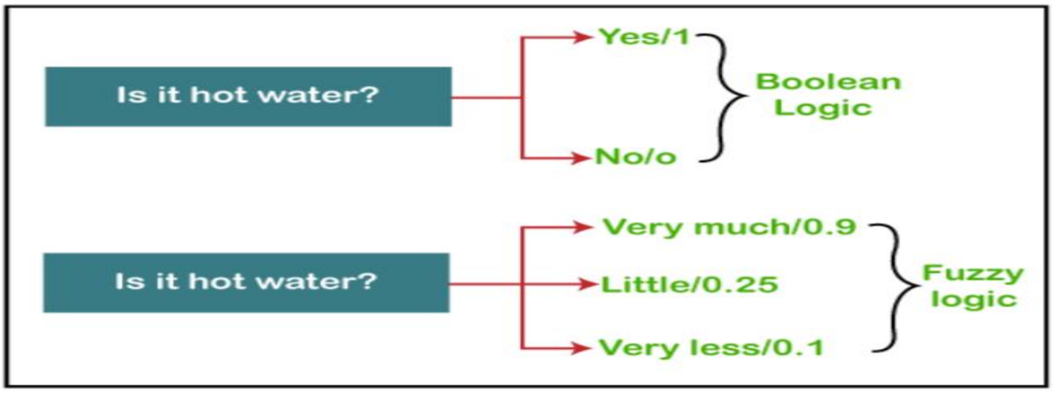
Aspect	Forward Chaining	Backward Chaining
Direction of Reasoning	Data-driven (starts with known facts)	Goal-driven (starts with the desired conclusion)
Process	Begins with initial facts and applies rules to derive new facts until a goal is reached.	Begins with a goal and works backward to determine if the known facts support it.
Usage	Useful for situations where all facts are known and the goal is to derive new knowledge.	Useful for goal-oriented queries where the goal needs to be proven.
Efficiency	Can generate a large number of conclusions and may be less efficient if many rules apply.	More efficient when searching for specific goals, as it narrows down to relevant rules.
Example	An expert system that updates knowledge based on new data inputs.	A medical diagnosis system that checks if symptoms lead to a specific diagnosis.

11	**What is the purpose of defuzzification in fuzzy logic reasoning?**	**K1**

- Defuzzification in fuzzy logic reasoning serves to convert fuzzy outputs into a precise value for practical use. It provides clear decisions or actions based on the results of fuzzy reasoning.
- This process simplifies complex fuzzy outputs, making them easier to interpret. In control applications, defuzzification generates exact control signals for systems like temperature regulation.
- Various techniques, such as the centroid method and mean of maxima, can be employed to achieve defuzzification based on specific requirements.
- Overall, it is essential for real-world applications that need definitive actions derived from fuzzy logic. By bridging the gap between fuzzy reasoning and actionable outcomes, defuzzification enhances decision-making processes.

12	**How does causal reasoning aid in diagnostic systems?**	**K2**

	• Causal reasoning aids diagnostic systems by identifying and establishing relationships between symptoms and underlying causes. • It helps generate hypotheses about potential conditions, guiding the diagnostic process effectively. • By making inferences based on known causal links, it prioritizes relevant tests and examinations. • This reasoning allows dynamic adjustments as new information becomes available, improving diagnostic accuracy. Ultimately, it enhances treatment planning by targeting root causes rather than just addressing symptoms	
13	**What is the goal of diagnosis reasoning in AI systems?**	**K1**
	• The goal of diagnostic reasoning in AI systems is to accurately identify and understand specific problems or malfunctions. • It aims to analyze the underlying causes of these issues for a comprehensive understanding. The system generates potential hypotheses based on the gathered evidence. • It provides actionable recommendations for resolving the identified problems. • Ultimately, it supports human experts in making informed decisions through insightful analyses and reasoning.	
14	**Mention an application of Diagnosis Reasoning.**	**K2**
	• Medical Diagnosis Systems • Fault Diagnosis in Engineering • Software Debugging • Network Security Systems • Automotive Diagnostic Systems	
15	**How is probabilistic reasoning used in diagnosis systems?**	**K2**
	• Probabilistic reasoning in diagnosis systems helps manage uncertainty by assigning probabilities to possible diagnoses based on incomplete information. • It often uses Bayesian networks to represent relationships between symptoms and conditions, updating probabilities as new evidence arises. • This approach calculates the likelihood of various diagnoses, helping to prioritize which conditions to investigate further. • It also assesses risks associated with different diagnoses, guiding recommendations for tests or treatments. • Ultimately, it supports healthcare professionals in making informed decisions, improving patient outcomes.	
	PART A	
S.NO	**QUESTIONS**	**BTL**
1	**Explain the importance of fuzzy sets in Fuzzy Logic.**	**K2**
	The 'Fuzzy' word means the things that are not clear or are vague. Sometimes, we cannot decide in real life that the given problem or statement is either true or false. At that time, this concept provides many values between the true and false and gives the flexibility to find the best solution to that problem. **Example of Fuzzy Logic as comparing to Boolean Logic**	



Fuzzy logic contains the multiple logical values and these values are the truth values of a variable or problem between 0 and 1. This concept was introduced by Lofti Zadeh in 1965 based on the Fuzzy Set Theory. This concept provides the possibilities which are not given by computers, but similar to the range of possibilities generated by humans.

In the Boolean system, only two possibilities (0 and 1) exist, where 1 denotes the absolute truth value and 0 denotes the absolute false value. But in the fuzzy system, there are multiple possibilities present between the 0 and 1, which are partially false and partially true.

The Fuzzy logic can be implemented in systems such as micro-controllers, workstation-based or large network-based systems for achieving the definite output. It can also be implemented in both hardware or software

Characteristics of Fuzzy Logic

Following are the characteristics of fuzzy logic:

1. This concept is flexible and we can easily understand and implement it.
2. It is used for helping the minimization of the logics created by the human.
3. It is the best method for finding the solution of those problems which are suitable for approximate or uncertain reasoning.
4. It always offers two values, which denote the two possible solutions for a problem and statement.
5. It allows users to build or create the functions which are non-linear of arbitrary complexity.
6. In fuzzy logic, everything is a matter of degree.
7. In the Fuzzy logic, any system which is logical can be easily fuzzified.
8. It is based on natural language processing.
9. It is also used by the quantitative analysts for improving their algorithm's execution.
10. It also allows users to integrate with the programming.

2	**Describe the components of a rule-based system with an example.**	**K2**

Rule based system in AI work:

Rule-based systems outline triggers & the actions that should follow (or are triggered). For example, a trigger might be an email containing the word “invoice”. An action might then be to forward the email to the finance team.

These rules most often take the form of "if" statements. "IF" outlines the trigger, "THEN" specifies the action to complete. So, if you want to create a rule-based system capable of handling 100 different actions, you’d have to write 100 different rules. If you want to then update the system and add actions, then you would need to write new rules.

In short, you use rules to tell a machine what to do, and the machine will do exactly as you tell it. From there, rule-based systems will execute the actions until you tell it to stop.
But remember: if you tell it to do something incorrectly, it will do it incorrectly.
The main components of a rules-based system:

A typical rule-based system has seven basic components:
The knowledge base
It holds the domain knowledge that is necessary for problem solving. In a rules-based system, the knowledge gets represented as a set of rules. Every rule specifies a relation, recommendation, directive, strategy or heuristic and has the IF (condition) THEN (action) structure. As soon as the condition part of the rule is satisfied, the rule gets triggered and the action part gets executed.
The database
The database has a set of facts that are used to compare against the IF (condition) part of the rules that are held in the knowledge base.
The inference engine
The inference engine is used to perform the reasoning through which the expert system comes to a solution. The job of the inference engine is to link the rules that are defined in the knowledge base with the facts that are stored in the database. The inference engine is also known as the semantic reasoner. It infers information or performs required actions on the basis of input and the rule base that's present in the knowledge base. The semantic reasoner involves a match-resolve-act cycle that works like this:

- **Match** - A section of the production rule system gets matched with the contents of the working memory to gain a conflict, where there are several instances of the satisfied productions.

- **Conflict-Resolution** - After the production system is matched, one of the production instances in the conflict is selected for execution for the purpose of determining the progress of the process.
- **Act** - The production instance selected in the previous stage is s executed, impacting the contents of the working memory.

Explanation facilities
The explanantion facilities make it possible for the user to ask the expert system how a specific conclusion was reached and why a specific fact is required. The expert system needs to be able to explain its reasoning and justify its advice, analysis, or conclusion.
User interface
The user interface is the means through which the user seeking a solution to a problem communicates with the expert system. The communication should be as meaningful and friendly as possible and the user interface should be as intuitive as possible.
These five elements are critical for any rule-based system. They are the core components of the rule-based system. But the system might have some additional components as well. A couple of these components could be the external interface and the working memory.
External interface
The external interface enables an expert system to work with external data files and programs that are written in conventional programming languages like C, Pascal, FORTRAN and Basic.

	Working memory The working memory stores temporary information and data.	
3	**Explain the concept of abductive reasoning with an example.**	**K2**
	Abductive reasoning is a form of logical inference that involves generating the best possible explanation for a set of observations or facts. It is often described as "inference to the best explanation" and is commonly used in everyday reasoning, problem-solving, and diagnostic systems. **Key Characteristics:** • **Starting Point**: Begins with incomplete or ambiguous data or observations. • **Goal**: Aims to find the most plausible explanation or hypothesis that accounts for the observed evidence. • **Flexibility**: Unlike deductive reasoning, which guarantees the truth of conclusions if the premises are true, abductive reasoning provides likely explanations that require further testing or verification. **Example of Abductive Reasoning:** **Scenario**: You walk into your kitchen and notice that the floor is wet. **Observations**: • The floor is wet. • You remember that you left the window open and it rained earlier. • There are no other visible leaks from the ceiling or plumbing. **Abductive Reasoning Process**: 1. **Hypothesis Generation**: Based on the observations, you might hypothesize that the wet floor is due to the rain coming in through the open window. 2. **Best Explanation**: While there could be other explanations (like someone spilling water or a leaky pipe), the most plausible and simplest explanation given the evidence is that rain entered through the open window. 3. **Further Investigation**: To confirm your hypothesis, you may check if the window is indeed open and look for any signs of water on the windowsill.	
4	**Discuss the role of Diagnosis Reasoning in industrial automation systems.**	**K3**
	Diagnosis reasoning plays a vital role in industrial automation systems by enabling efficient monitoring, troubleshooting, and maintenance of automated processes. Here's how it contributes: 1. **Fault Detection**: Diagnosis reasoning helps identify and detect faults or anomalies in machinery and production processes, allowing for early intervention before they lead to significant downtime or damage. 2. **Root Cause Analysis**: By analyzing symptoms and operational data, diagnosis reasoning can pinpoint the root causes of failures, enabling engineers to address underlying issues rather than just the symptoms. 3. **Predictive Maintenance**: It supports predictive maintenance strategies by assessing the condition of equipment and predicting when failures might occur, allowing for timely maintenance and reducing unplanned downtime. 4. **Decision Support**: Diagnosis reasoning provides valuable insights and recommendations for operators and engineers, helping them make informed decisions about repairs, replacements, or adjustments needed to optimize performance.	

	5. **Process Optimization**: By continuously monitoring system performance and diagnosing inefficiencies, it aids in optimizing production processes, improving overall efficiency and productivity. 6. **Safety Assurance**: In industrial environments, diagnosis reasoning enhances safety by detecting potential hazards and enabling timely responses to prevent accidents or unsafe conditions. 7. **Integration with Control Systems**: Diagnosis reasoning can be integrated with control systems to automate responses to detected faults, improving the system's resilience and reducing the need for human intervention.	
5	**Explain the properties of classical set with an example.**	**K4**
	Properties of Classical Set There are following various properties which play an essential role for finding the solution of a fuzzy logic problem. **1. Commutative Property:** This property provides the following two states which are obtained by two finite sets A and B: $A \cup B = B \cup A$ $A \cap B = B \cap A$ **2. Associative Property:** This property also provides the following two states but these are obtained by three different finite sets A, B, and C: $A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$ **3. Idempotency Property:** This property also provides the following two states but for a single finite set A: $A \cup A = A$ $A \cap A = A$ **4. Absorption Property** This property also provides the following two states for any two finite sets A and B: $A \cup (A \cap B) = A$ $A \cap (A \cup B) = A$ **5. Distributive Property:** This property also provides the following two states for any three finite sets A, B, and C: $A\cup (B \cap C) = (A \cup B)\cap (A \cup C)$ $A\cap (B \cup C) = (A\cap B) \cup (A\cap C)$ **6. Identity Property:** This property provides the following four states for any finite set A and Universal set X: $A \cup \varphi =A$ $A \cap X = A$ $A \cap \varphi = \varphi$ $A \cup X = X$ **7. Transitive property** This property provides the following state for the finite sets A, B, and C: If $A \subseteq B \subseteq C$, then $A \subseteq C$ **8. Ivolution property** This property provides following state for any finite set A:	

$$\boxed{\overline{\overline{A}} = A}$$

9. De Morgan's Law

This law gives the following rules for providing the contradiction and tautologies:

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

PART C

S.NO	QUESTIONS	BTL
1	**Compare and contrast deductive, inductive, and abductive reasoning with suitable examples.**	K3

Aspect	Deductive Reasoning	Inductive Reasoning	Abductive Reasoning
Definition	Draws specific conclusions from general principles or premises.	Derives general conclusions from specific observations.	Infers the most likely explanation for a set of observations.
Certainty of Conclusion	Guarantees certainty if premises are true.	Provides probable conclusions based on evidence.	Suggests plausible conclusions, requiring further investigation.
Direction of Reasoning	Top-down (general to specific).	Bottom-up (specific to general).	Best explanation for a given set of facts.
Logical Structure	Follows a clear logical structure (if premises, then conclusion).	Involves pattern recognition and generalization.	Involves creating hypotheses based on the best available explanation.
Use of Evidence	Relies on accepted truths or established premises.	Based on observed data and experiences.	Based on observations and existing knowledge but requires assumption.
Risk of Error	Low risk of error if premises are valid; conclusion is certain.	Higher risk of error, as conclusions may not apply universally.	Moderate risk of error; conclusions may need further testing.
Example 1	All mammals have hearts. A dog is a mammal. Therefore, a dog has a heart.	Every time I've seen a car, it was parked outside my house. Therefore, cars are usu[illegible] parked outside my house.	The smoke alarm is going off. The most likely explanation is that there is a fire in the building.

Example 2	All birds have feathers. A sparrow is a bird. Therefore, a sparrow has feathers.	The sun has risen in the east every day of my life. Therefore, the sun will rise in the east tomorrow.	You find your keys missing. The best explanation is that you left them in the car.
Applications	Used in mathematics, formal logic, legal reasoning, and certain fields of science.	Commonly used in scientific research, data analysis, market research, and forming hypotheses.	Frequently used in diagnostic reasoning (medicine, troubleshooting), forensic analysis, and everyday decision-making.
Strengths	Provides clear and definitive conclusions; high reliability when premises are accurate.	Useful for discovering patterns, making predictions, and generating new theories.	Helps in situations with incomplete information; good for generating hypotheses.
Limitations	Depends heavily on the validity of the premises; may not apply in complex real-world situations.	Conclusions may be overly broad and not universally applicable; susceptible to biases.	Conclusions may not be accurate; relies on the assumption that the most plausible explanation is the correct one.

2	**Explain about the reasoning and its types with example.**	**K2**

The reasoning is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. Or we can say, "Reasoning is a way to infer facts from existing data." It is a general process of thinking rationally, to find valid conclusions.

In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

Types of Reasoning

In artificial intelligence, reasoning can be divided into the following categories:

o Deductive reasoning
o Inductive reasoning
o Abductive reasoning
o Common Sense Reasoning
o Monotonic Reasoning
o Non-monotonic Reasoning

1. Deductive reasoning:

Deductive reasoning is deducing new information from logically related known information. It is the form of valid reasoning, which means the argument's conclusion must be true when the premises are true.

Deductive reasoning is a type of propositional logic in AI, and it requires various rules and facts.

It is sometimes referred to as top-down reasoning, and contradictory to inductive reasoning.

In deductive reasoning, the truth of the premises guarantees the truth of the conclusion.

Deductive reasoning mostly starts from the general premises to the specific conclusion, which can be explained as below example.

Example:

Premise-1: All the human eats veggies

Premise-2: Suresh is human.
Conclusion: Suresh eats veggies.
The general process of deductive reasoning is given below:

2. Inductive Reasoning:
Inductive reasoning is a form of reasoning to arrive at a conclusion using limited sets of facts by the process of generalization. It starts with the series of specific facts or data and reaches to a general statement or conclusion.
Inductive reasoning is a type of propositional logic, which is also known as cause-effect reasoning or bottom-up reasoning.
In inductive reasoning, we use historical data or various premises to generate a generic rule, for which premises support the conclusion.
In inductive reasoning, premises provide probable supports to the conclusion, so the truth of premises does not guarantee the truth of the conclusion.
Example:

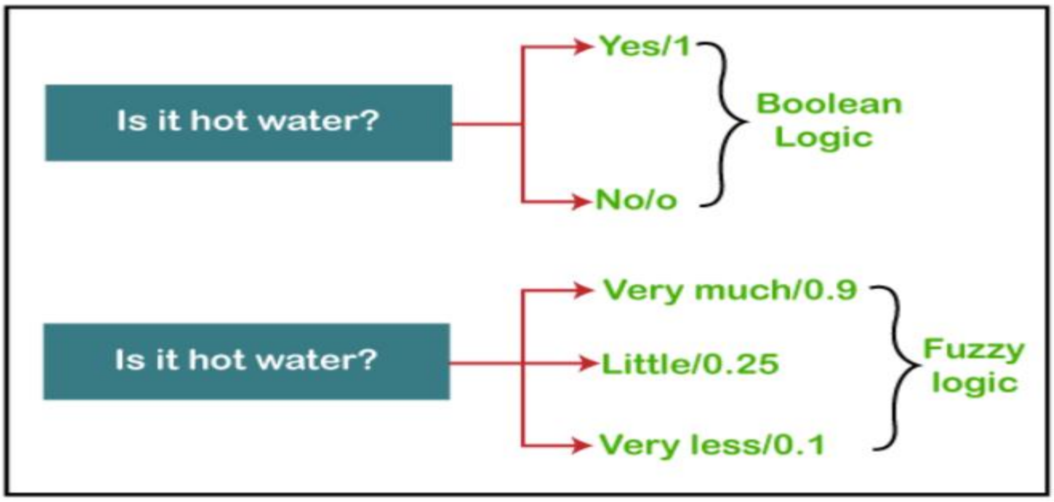
Premise: All of the pigeons we have seen in the zoo are white.
Conclusion: Therefore, we can expect all the pigeons to be white.

3. Abductive reasoning:
Abductive reasoning is a form of logical reasoning which starts with single or multiple observations then seeks to find the most likely explanation or conclusion for the observation.
Abductive reasoning is an extension of deductive reasoning, but in abductive reasoning, the premises do not guarantee the conclusion.
Example:
Implication: Cricket ground is wet if it is raining
Axiom: Cricket ground is wet.
Conclusion It is raining.
4. Common Sense Reasoning
Common sense reasoning is an informal form of reasoning, which can be gained through experiences.
Common Sense reasoning simulates the human ability to make presumptions about events which occurs on every day.
It relies on good judgment rather than exact logic and operates on heuristic knowledge and heuristic rules.
Example:
1. One person can be at one place at a time.
2. If I put my hand in a fire, then it will burn.
The above two statements are the examples of common-sense reasoning which a human mind can easily understand and assume.

5. Monotonic Reasoning:
In monotonic reasoning, once the conclusion is taken, then it will remain the same even if we add some other information to existing information in our knowledge base. In monotonic reasoning, adding knowledge does not decrease the set of prepositions that can be derived.

	To solve monotonic problems, we can derive the valid conclusion from the available facts only, and it will not be affected by new facts. Monotonic reasoning is not useful for the real-time systems, as in real time, facts get changed, so we cannot use monotonic reasoning. Monotonic reasoning is used in conventional reasoning systems, and a logic-based system is monotonic. Any theorem proving is an example of monotonic reasoning. **Example:** **o Earth revolves around the Sun.** **It is a true fact, and it cannot be changed even if we add another sentence in knowledge base like,** **"The moon revolves around the earth" Or "Earth is not round," etc.** **Advantages of Monotonic Reasoning:** o In monotonic reasoning, each old proof will always remain valid. o If we deduce some facts from available facts, then it will remain valid for always. **Disadvantages of Monotonic Reasoning:** o We cannot represent the real world scenarios using Monotonic reasoning. o Hypothesis knowledge cannot be expressed with monotonic reasoning, which means facts should be true. o Since we can only derive conclusions from the old proofs, so new knowledge from the real world cannot be added. **6. Non-monotonic Reasoning** In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base. Logic will be said as non-monotonic if some conclusions can be invalidated by adding more knowledge into our knowledge base. Non-monotonic reasoning deals with incomplete and uncertain models. "Human perceptions for various things in daily life, "is a general example of non-monotonic reasoning. **Example: Let suppose the knowledge base contains the following knowledge:** **o Birds can fly** **o Penguins cannot fly** **o Pitty is a bird** So from the above sentences, we can conclude that Pitty can fly. However, if we add one another sentence into knowledge base "Pitty is a penguin", which concludes "Pitty cannot fly", so it invalidates the above conclusion. **Advantages of non-monotonic reasoning:** o For real-world systems such as Robot navigation, we can use non-monotonic reasoning. o In Non-monotonic reasoning, we can choose probabilistic facts or can make assumptions. **Disadvantages of Non-monotonic Reasoning:** o In non-monotonic reasoning, the old facts may be invalidated by adding new sentences. o It cannot be used for theorem proving.	
3	**Explain the architecture of a Fuzzy Logic control system with an industrial application example.**	**K2**
	The 'Fuzzy' word means the things that are not clear or are vague. Sometimes, we cannot decide in real life that the given problem or statement is either true or false. At that time,	

this concept provides many values between the true and false and gives the flexibility to find the best solution to that problem. **Example of Fuzzy Logic as comparing to Boolean Logic**



Fuzzy logic contains the multiple logical values and these values are the truth values of a variable or problem between 0 and 1. This concept was introduced by Lofti Zadeh in 1965 based on the Fuzzy Set Theory. This concept provides the possibilities which are not given by computers, but similar to the range of possibilities generated by humans.

In the Boolean system, only two possibilities (0 and 1) exist, where 1 denotes the absolute truth value and 0 denotes the absolute false value. But in the fuzzy system, there are multiple possibilities present between the 0 and 1, which are partially false and partially true.

The Fuzzy logic can be implemented in systems such as micro-controllers, workstation-based or large network-based systems for achieving the definite output. It can also be implemented in both hardware or software

Characteristics of Fuzzy Logic

Following are the characteristics of fuzzy logic:

1. This concept is flexible and we can easily understand and implement it.
2. It is used for helping the minimization of the logics created by the human.
3. It is the best method for finding the solution of those problems which are suitable for approximate or uncertain reasoning.
4. It always offers two values, which denote the two possible solutions for a problem and statement.
5. It allows users to build or create the functions which are non-linear of arbitrary complexity.
6. In fuzzy logic, everything is a matter of degree.
7. In the Fuzzy logic, any system which is logical can be easily fuzzified.
8. It is based on natural language processing.
9. It is also used by the quantitative analysts for improving their algorithm's execution.
10. It also allows users to integrate with the programming.

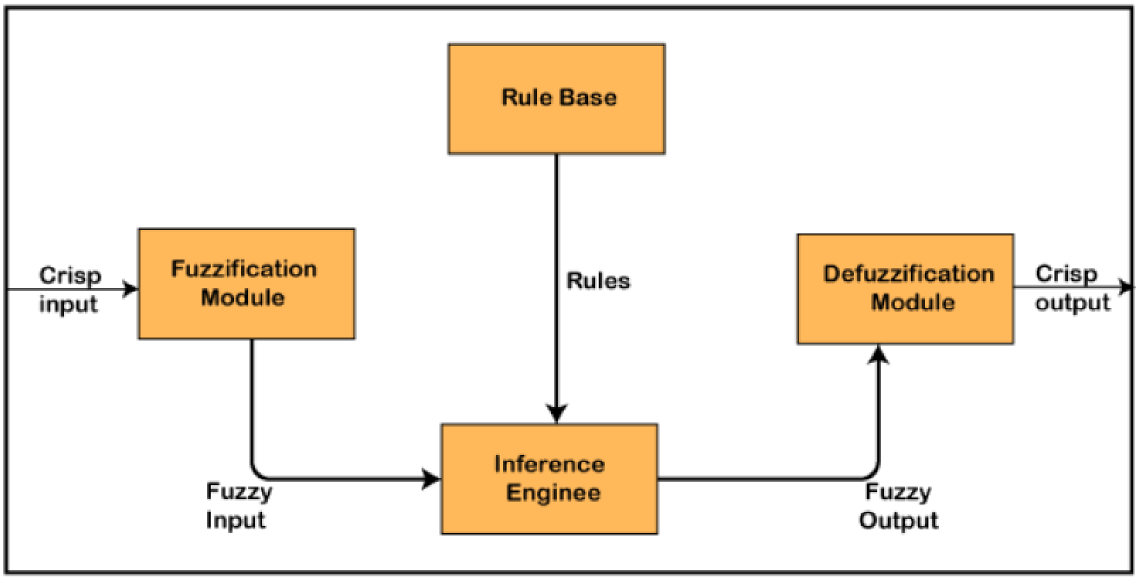
Architecture of a Fuzzy Logic System In the architecture of the Fuzzy Logic system, each component plays an important role.

The architecture consists of the different four components which are given below.

1. Rule Base

2. Fuzzification
3. Inference Engine
4. Defuzzification

Following diagram shows the architecture or process of a Fuzzy Logic system:



1. Rule Base

Rule Base is a component used for storing the set of rules and the If-Then conditions given by the experts are used for controlling the decision-making systems. There are so many updates that come in the Fuzzy theory recently, which offers effective methods for designing and tuning of fuzzy controllers. These updates or developments decreases the number of fuzzy set of rules.

2. Fuzzification

Fuzzification is a module or component for transforming the system inputs, i.e., it converts the crisp number into fuzzy steps. The crisp numbers are those inputs which are measured by the sensors and then fuzzification passed them into the control systems for further processing. This component divides the input signals into following five states in any Fuzzy Logic system:

o Large Positive (LP)
o Medium Positive (MP)
o Small (S)
o Medium Negative (MN)
o Large negative (LN)

3. Inference Engine

This component is a main component in any Fuzzy Logic system (FLS), because all the information is processed in the Inference Engine. It allows users to find the matching degree between the current fuzzy input and the rules. After the matching degree, this system determines which rule is to be added according to the given input field. When all rules are fired, then they are combined for developing the control actions.

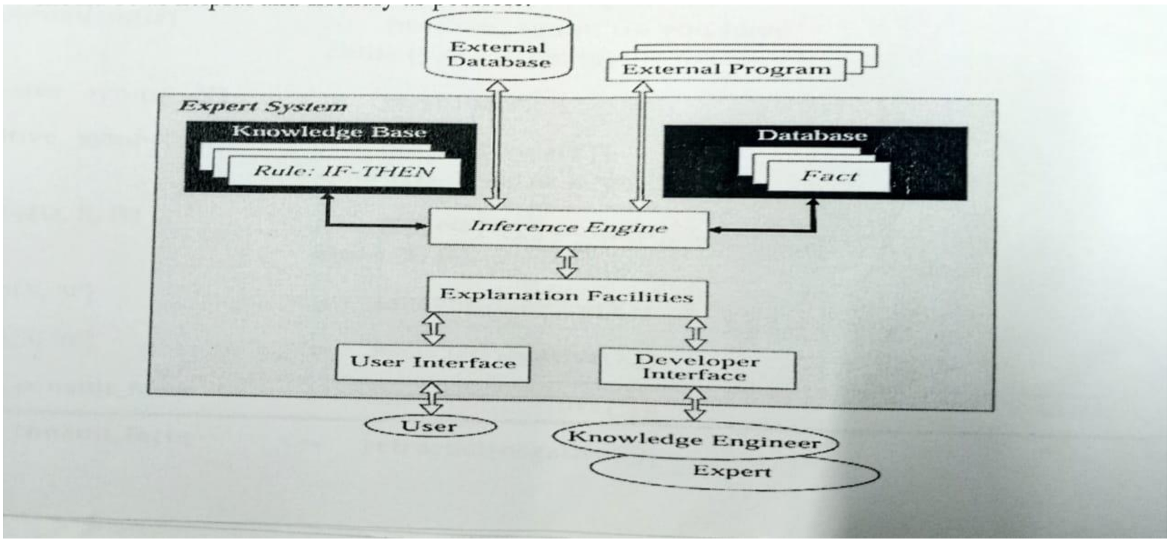
4. Defuzzification

Defuzzification is a module or component, which takes the fuzzy set inputs generated by the Inference Engine, and then transforms them into a crisp value. It is the last step in the process of a fuzzy logic system. The crisp value is a type of value which is acceptable by the user. Various techniques are present to do this, but the user has to select the best one for reducing the errors.

	Membership Function The membership function is a function which represents the graph of fuzzy sets, and allows users to quantify the linguistic term. It is a graph which is used for mapping each element of x to the value between 0 and 1. This function is also known as indicator or characteristics function. This function of Membership was introduced in the first papers of fuzzy set by Zadeh. For the Fuzzy set B, the membership function for X is defined as: $\mu B:X \rightarrow [0,1]$. In this function X, each element of set B is mapped to the value between 0 and 1. This is called a degree of membership or membership value	
4	**Describe the architecture of a rule-based reasoning system, explaining its main components.**	**K2**
	A rule-based system is a system that applies human-made rules to store, sort and manipulate data. In doing so, it mimics human intelligence. Rule-based systems require a set of facts or source of data, and a set of rules for manipulating that data. These rules are sometimes referred to as 'If statements' as they tend to follow the line of 'IF X happens THEN do Y'. The steps can be simplified to: • First comes the data or new business event • Then comes the analysis: the part where the system conditionally processes the data against its rules • Then comes any subsequent automated follow-up actions **Some of the important elements of rule-based systems include:** **A set of facts** These facts are assertions or anything that is relevant to the beginning state of the system. **Set of Rules** This set contains all the actions that should be performed within the scope of a problem and defines how to act on the assertion set. In the set of rules facts are represented in an IF-THEN form. **Termination Criteria or Interpreter** This determines whether a solution exists or not and figures out when the process should be terminated **Characteristics of rule-based systems: Some of the features of rule-based systems are:** • They are made up of the combined knowledge of human experts in the problem domain. • They represent knowledge in a very declarative manner. • They make it possible to use various knowledge representations paradigms. • They support the implementation of non-deterministic search and control strategies. • They help in describing fragmentary, ill-structured, heuristic, judgemental knowledge. • They are robust and have the ability to operate using uncertain or incomplete knowledge. • They can help with rule-based decision making **Rule based system in AI work:** Rule-based systems outline triggers & the actions that should follow (or are triggered). For example, a trigger might be an email containing the word "invoice". An action might then be to forward the email to the finance team.	

These rules most often take the form of "if" statements. "IF" outlines the trigger, "THEN" specifies the action to complete. So, if you want to create a rule-based system capable of handling 100 different actions, you'd have to write 100 different rules. If you want to then update the system and add actions, then you would need to write new rules.
In short, you use rules to tell a machine what to do, and the machine will do exactly as you tell it. From there, rule-based systems will execute the actions until you tell it to stop.
But remember: if you tell it to do something incorrectly, it will do it incorrectly.

The main components of a rules-based system:



A typical rule-based system has seven basic components:

The knowledge base

It holds the domain knowledge that is necessary for problem solving. In a rules-based system, the knowledge gets represented as a set of rules. Every rule specifies a relation, recommendation, directive, strategy or heuristic and has the IF (condition) THEN (action) structure. As soon as the condition part of the rule is satisfied, the rule gets triggered and the action part gets executed.

The database

The database has a set of facts that are used to compare against the IF (condition) part of the rules that are held in the knowledge base.

The inference engine

The inference engine is used to perform the reasoning through which the expert system comes to a solution. The job of the inference engine is to link the rules that are defined in the knowledge base with the facts that are stored in the database. The inference engine is also known as the semantic reasoner. It infers information or performs required actions on the basis of input and the rule base that's present in the knowledge base. The semantic reasoner involves a match-resolve-act cycle that works like this:

- **Match** - A section of the production rule system gets matched with the contents of the working memory to gain a conflict, where there are several instances of the satisfied productions.

	• **Conflict-Resolution -** After the production system is matched, one of the production instances in the conflict is selected for execution for the purpose of determining the progress of the process. • **Act** - The production instance selected in the previous stage is s executed, impacting the contents of the working memory. **Explanation facilities** The explanantion facilities make it possible for the user to ask the expert system how a specific conclusion was reached and why a specific fact is required. The expert system needs to be able to explain its reasoning and justify its advice, analysis, or conclusion. **User interface** The user interface is the means through which the user seeking a solution to a problem communicates with the expert system. The communication should be as meaningful and friendly as possible and the user interface should be as intuitive as possible. These five elements are critical for any rule-based system. They are the core components of the rule-based system. But the system might have some additional components as well. A couple of these components could be the external interface and the working memory. **External interface** The external interface enables an expert system to work with external data files and programs that are written in conventional programming languages like C, Pascal, FORTRAN and Basic.	
5	**Explain diagnosis reasoning in AI and its application in fault detection and troubleshooting.**	**K2**
	Diagnosis reasoning in AI refers to the process of using artificial intelligence to identify, understand, and resolve problems or malfunctions in a system. This type of reasoning is typically applied in domains like healthcare, manufacturing, automotive, telecommunications, and more. Diagnosis reasoning involves the use of AI techniques to analyze data from systems, infer the root causes of problems, and suggest corrective actions. This is essential in fault detection and troubleshooting, where the goal is to quickly and accurately identify faults and take appropriate action to fix them. **Key Components of Diagnosis Reasoning in AI:** 1. **Knowledge Representation:** o Systems require a well-organized database of rules, facts, and models about the system to be diagnosed. This knowledge can be represented in various forms like decision trees, Bayesian networks, or rule-based systems. 2. **Inference Mechanism:** o Inference mechanisms are used to apply knowledge to specific problems. They can use deductive reasoning (top-down), abductive reasoning (hypothesize a possible cause and then test), or model-based reasoning (comparing observed behavior to a theoretical model). 3. **Data Collection and Analysis:** o Continuous monitoring of sensor data, system logs, or other diagnostic inputs is needed for fault detection. Machine learning algorithms, pattern recognition, or anomaly detection can help analyze the data. 4. **Probabilistic Reasoning:** o Many AI systems use probabilistic models (like Bayesian networks) to reason about uncertainty, identifying the most likely causes of a failure or issue based on observed symptoms. 5. **Automated Decision-Making:**	

- Once faults are detected and diagnosed, AI systems can also suggest possible solutions or recommend actions to mitigate the problem.

Applications in Fault Detection and Troubleshooting:

1. **Manufacturing:**
    - AI is widely used in fault detection for manufacturing equipment and industrial machinery. Sensors are used to monitor equipment conditions (e.g., temperature, vibrations), and AI algorithms predict when machinery might fail (predictive maintenance) or detect current faults that need immediate attention (reactive maintenance).
    - Example: Detecting anomalies in the functioning of an assembly line robot and diagnosing potential mechanical failures before they cause production downtime.
2. **Automotive:**
    - AI-powered diagnostic tools are used in vehicles to detect engine or component faults in real-time. For example, modern cars use onboard diagnostic systems that provide alerts to the driver about issues like fuel inefficiency, emission problems, or mechanical faults.
    - Example: Tesla's AI-based vehicle diagnostics system that continuously monitors sensor data and suggests repairs or software updates remotely.