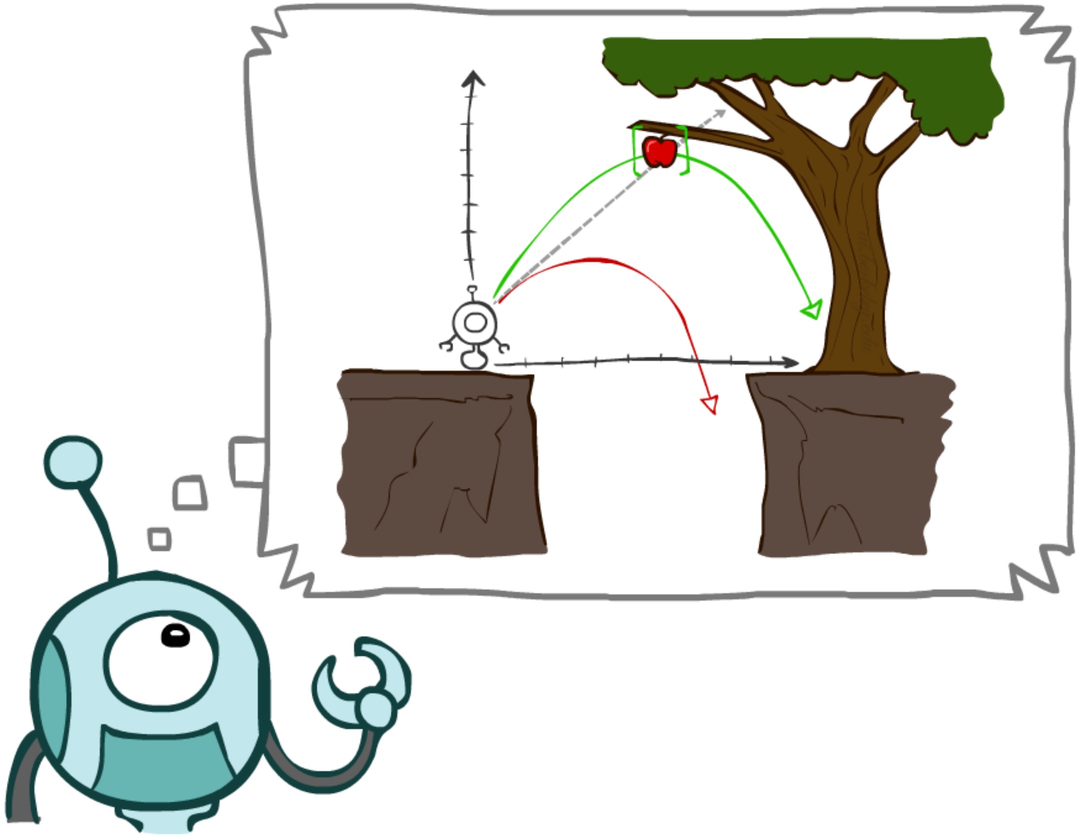
Fall 2024

BECE309L – AI & ML

Knowledge Representation

Module – 3

Dr. Nitish Katal



Outline

- Knowledge-based agents
- Agents based on Propositional Logic
- First-order logic

Terminologies

Knowledge base (KB):

- Key component of a knowledge-based agent.
- These deal with **real facts of world**.
- It is a mixture of sentences which are explained in knowledge representation language.

Inference Engine(IE):

- It is **knowledge-based system** engine used to **infer new knowledge** in the system.

Knowledge Representation (KR):

- The field in AI that focuses on how to **formally represent and manage knowledge about the world** in a way that allows **computers to use it effectively** for **reasoning** and **decision-making**.

Ontologies:

- **Formal representations** that **define a set of concepts and the relationships between them** within a domain, **providing** a **shared vocabulary** and a **structure** for **reasoning**.

Terminologies

Knowledge Representation Structures

Knowledge Base (KB):

- A **collection of knowledge representations** that an **AI system uses** to **perform reasoning** tasks.

Taxonomy:

- A **hierarchical classification** of **concepts** or **entities**, often used within ontologies.

Frames:

- **Data structures** that **represent stereotypical situations** or **objects** by **including attributes** and **values**.
- Each **frame** is like an **object** with **specific properties**.

Terminologies

Inference and Reasoning:

- Inference:
  - The ***process*** of ***deriving*** new ***information*** or ***conclusions*** from ***existing knowledge***.

- Reasoning
  - Deductive Reasoning:
    - Drawing **specific conclusions** from **general principles** or **premises**.
  - Inductive Reasoning:
    - Generalizing from **specific instances** to **broader principles**.
  - Abductive Reasoning:
    - Inferring the **best explanation** for a **set** of **observations** or **facts**.

Terminologies

Examples : Inference and Reasoning:

- Deductive Reasoning
  - Premise 1: All mammals have a backbone.
  - Premise 2: A whale is a mammal.
  - Conclusion: Therefore, a whale has a backbone.
  - In this example, the conclusion is a specific fact drawn from the general principle that all mammals have a backbone.
- Inductive Reasoning
  - Example: You observe that: The sun has risen in the east every morning so far.
  - It rose in the east yesterday.
  - It rose in the east today.
  - Conclusion: Therefore, the sun will likely rise in the east tomorrow.
  - Here, the conclusion generalizes a pattern based on specific observations.
- Abductive Reasoning
  - Example: Observation: The grass is wet in the morning.
  - Possible Explanation 1: It rained during the night.
  - Possible Explanation 2: The sprinkler was left on overnight.
  - Conclusion: Since there are no signs of rainfall and the sprinkler was indeed on, the best explanation is that the sprinkler caused the wet grass.
  - Abductive reasoning involves inferring the most likely explanation from the available evidence.

Terminologies

Types of Knowledge

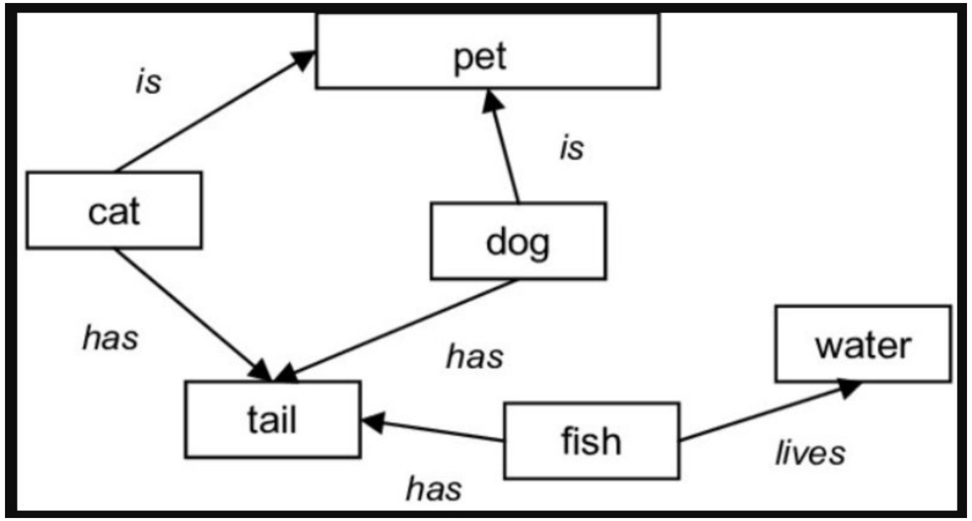
- Declarative Knowledge:
  - **Knowledge** about **facts** and **information**
  - e.g., "*The Eiffel Tower is in Paris*".
- Procedural Knowledge:
  - Knowledge about **how to perform tasks** or procedures
  - e.g., "*How to change a tire*".

Representation Models

- Propositional Logic:
  - A **form of logic** where **statements** are **either true or false**, and
  - **Logical operations** are **performed** on these **statements**.
- Predicate Logic:
  - An **extension of propositional logic** that **includes** **predicates** (functions that return true or false) and **quantifiers** (e.g., "For all x" or "There exists an x").

Terminologies

- Semantic Networks:
  - **Graph-based structures** where **nodes** represent **entities** or **concepts** and **edges** represent **relationships** between **them**.
- Semantic relationships
- Hierarchy:
  - A relationship where one concept is a more general or more specific instance of another (e.g., "Dog" is a more specific instance of "Animal").
- Association:
  - A relationship where concepts are related but not hierarchically (e.g., "Doctor" is associated with "Hospital").



Terminologies

- Uncertainty and Ambiguity

- Probabilistic Reasoning:
  - A **method** of **dealing** with **uncertainty** by using **probability theory** to **represent** and **infer** the **likelihood** of **various outcomes**.
  - *Example*:
  - Imagine you're a doctor diagnosing a patient with symptoms of fever, cough, and fatigue. You know that there are several possible causes, such as the flu, COVID-19, or a common cold. Using probabilistic reasoning, you assign probabilities based on the prevalence of these illnesses in the community:
    - *Probability of the flu: 50%*
    - *Probability of COVID-19: 30%*
    - *Probability of the common cold: 20%*
  - Given the symptoms and their likelihood, you might infer that the flu is the most likely cause.

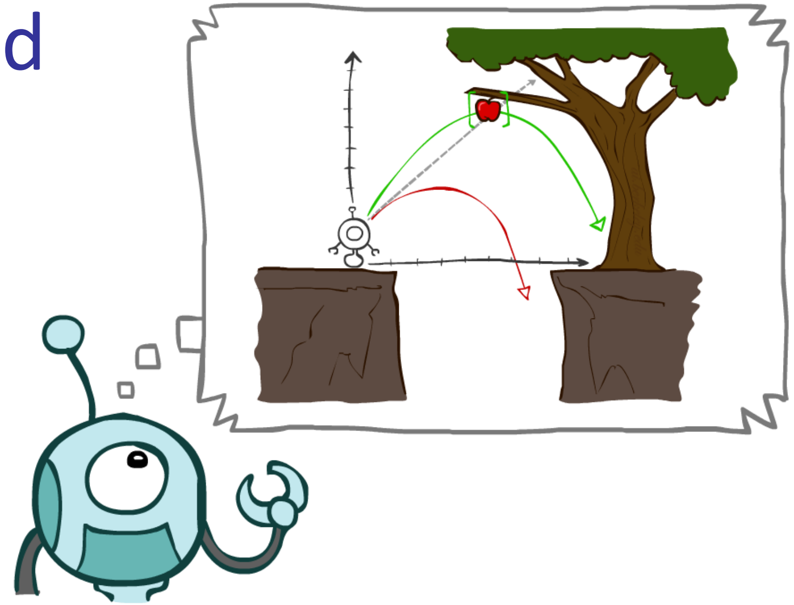
Terminologies

- Knowledge Acquisition and Learning

- Knowledge Acquisition:
  - The **process** of **gathering** and **structuring knowledge** from **various sources** to **populate** a **knowledge base**.
  - *Example:*
  - A software company is developing an expert system to assist with medical diagnoses. To populate the system with knowledge, they interview several doctors and gather information from medical textbooks and research papers.
  - They extract rules like:
    - *"If the patient has a high fever and cough, consider testing for the flu.“*
    - *"If the patient has loss of taste and smell, test for COVID-19.“*
  - This gathered and structured knowledge is then used to create a knowledge base that the expert system relies on to make informed decisions.

Agents that know things

- Agents acquire knowledge through perception, learning, language
  - Knowledge of the *effects of actions* (“transition model”)
  - Knowledge of how the *world affects sensors* (“sensor model”)
  - *Knowledge of the current state of the world*
- Can keep track of a partially observable world
- Can formulate plans to achieve goals



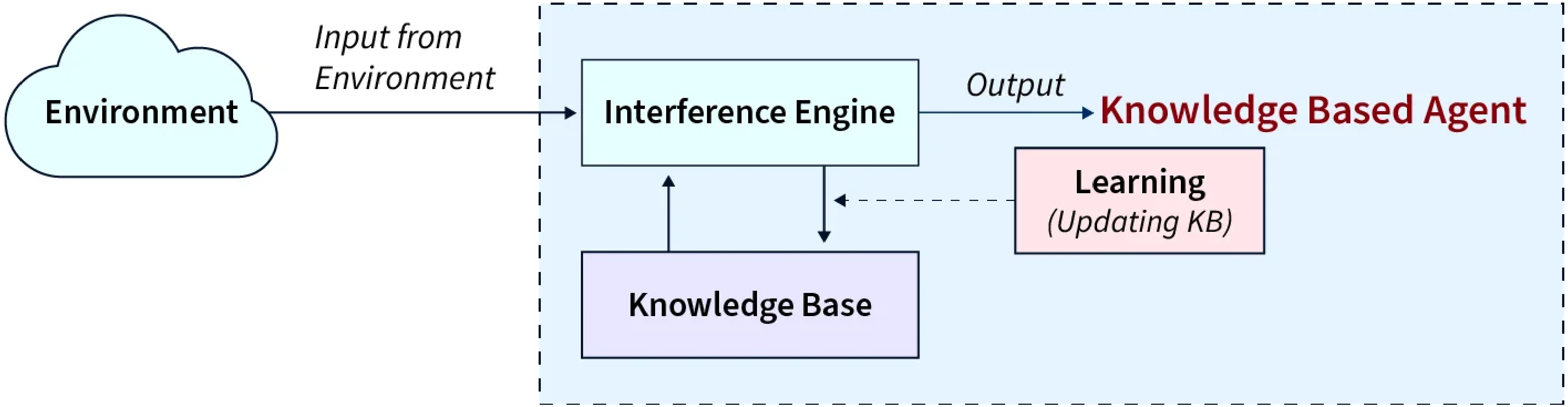
Knowledge

- Knowledge base = set of sentences in a formal language
- Declarative approach to building an agent (or other system):
  - ***Tell*** it what it needs to know (or have it ***Learn*** the knowledge)
  - Then it can ***Ask*** itself what to do—answers should follow from the KB
- Agents can be viewed at the ***knowledge level***
  i.e., what they ***know***, regardless of how implemented
- A single inference algorithm can answer any answerable question

Knowledge base	Domain-specific facts
Inference engine	Generic code

Knowledge based Agents

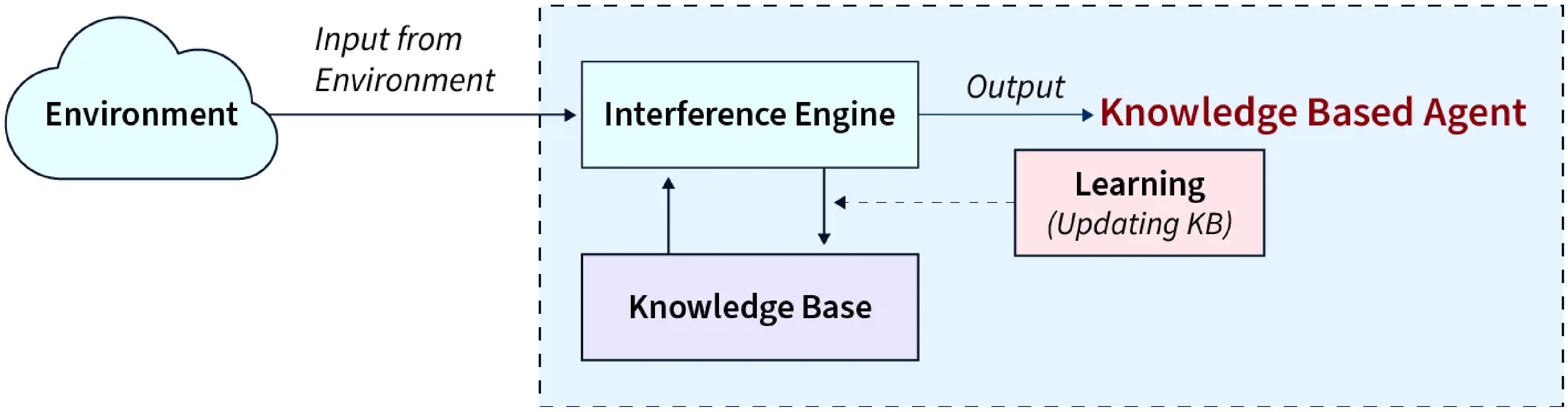
- KBAs are **artificial intelligence systems** that use **knowledge** to **perform their tasks**.
- For example, they make ***deductions***, ***decisions***, and ***conclusions*** based on ***available knowledge***.



Knowledge based Agents

- Architecture

- KBA receives input from the environment through perception, which the inference engine processes.
- The inference engine communicates with the knowledge base (KB) to determine the appropriate action based on the knowledge stored in the KB.
- The *learning element* of the *KBA regularly updates* the *KB* by *incorporating new knowledge*.

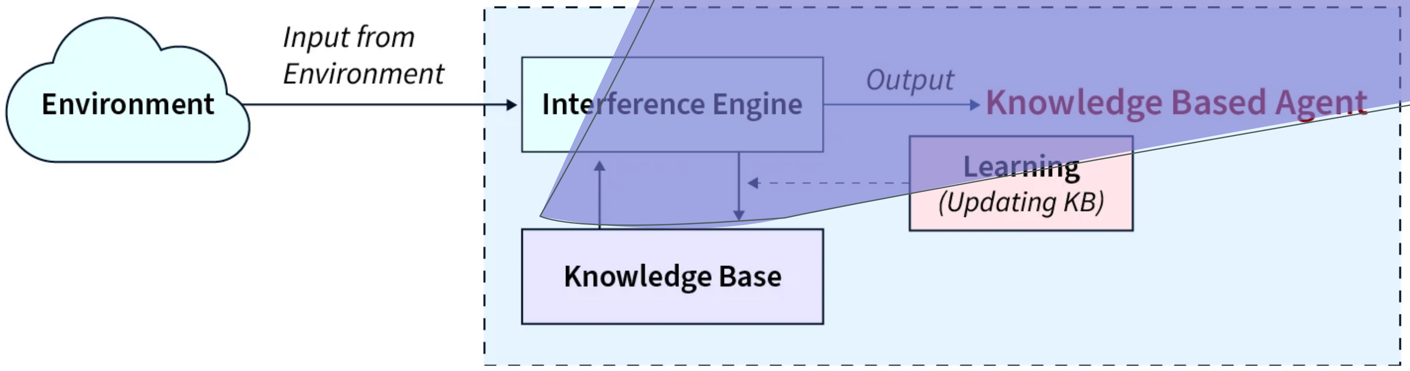


Knowledge based Agents

- KBA's are composed of 2 main modules:
  - Knowledge Base
  - Inference Engine

Knowledge Base {KB}

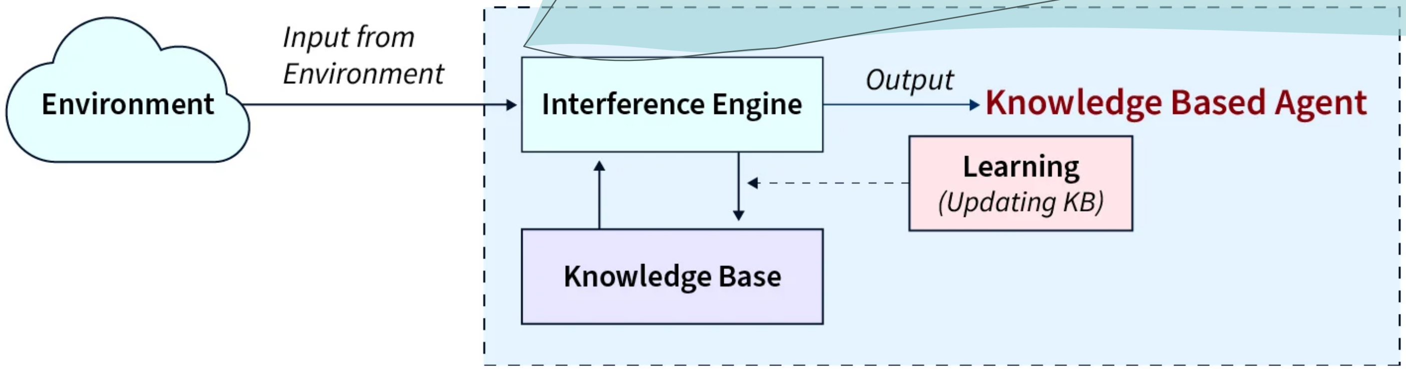
- ❑ KB of a KBA is a **collection of knowledge** that the **agent** uses to **make decisions**.
- ❑ The **knowledge can be explicit**, such as **rules** or **facts**, or **implicit**, such as **relationships** or **patterns**.
- ❑ KB is **stored** in a **database** or a knowledge representation system.



Knowledge based Agents

Inference Engine {IE}

- ❑ **Inference system** of a KBA is **responsible** for **using the knowledge** in the **knowledge base** to **make decisions**.
- ❑ IE uses **reasoning methods**, such as **deduction**, **induction**, and **abduction**, to **infer new knowledge** from existing knowledge.
- ❑ **Generating new facts** by the **inference system** allows the agent to **update its knowledge base**.
- ❑ **Operates primarily through two rules**: **forward chaining** and **backward chaining**.



Knowledge based Agents

- **Forward Chaining:**
  - Forward Chaining the Inference Engine goes through all the facts, conditions and derivations before deducing the outcome
  - When based on available data a decision is taken then the process is called as **Forwarding chaining**.
  - It works from an initial state and reaches to the goal(final decision).
  - Example:
    - A
    - A -> B
    - B

    or
    - He is running.
    - If he is running, he sweats.
    - He is sweating.

- **Backward Chaining:**
  - In this, the inference system knows the final decision or goal.
  - This system starts from the goal and works backwards to determine what facts must be asserted so that the goal can be achieved,
  - It works from goal(final decision) and reaches the initial state.
  - Example:
    - B
    - A -> B
    - A

    or
    - He is sweating.
    - If he is running, he sweats.
    - He is running.

Knowledge based Agents

- KBA *receives* a percept and *produces* an action as output.
  - The agent is equipped with a knowledge base (KB) with background knowledge of the real world.
  - Additionally, the agent has a time counter initially set to zero, *indicating the time elapsed during the process*.

- Each *time the function is executed*, the agent performs three operations:
  - Firstly, it *reports to the KB* what it has perceived.
  - Secondly, *it asks the KB* what action it should take.
  - Thirdly, *it reports to the KB* which action it has chosen.

Knowledge based Agents

- MAKE-PERCEPT-SENTENCE
  - Generates a sentence that indicates that the agent perceived the given percept at the given time.
- MAKE-ACTION-QUERY
  - Generates a sentence that asks which action should be taken at the current time.
- MAKE-ACTION-SENTENCE
  - Generates a sentence that asserts that the chosen action has been executed.

Knowledge based Agents

- KBA 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.

Levels of KBAs

- Knowledge-based agents can be classified into three levels:

- Knowledge Level
  - Knowledge level is the **highest level** of abstraction in a knowledge-based agent.
  - It describes what the agent knows and how it uses it to perform tasks.
  - Knowledge level concerns the *representation* and *organization* of *knowledge* rather than the *implementation details*.
  - *Example*, suppose an automated taxi agent needs to go from a station A to station B, and it knows the way from A to B, so this comes at the knowledge level.

- Logical Level
  - Logical level is the **intermediate level** of abstraction in a knowledge-based agent.
  - It describes how knowledge is represented and manipulated by inference engine.
  - Logical story concerns the formal logic used to represent knowledge and make inferences.
  - *Example*: At the logical level we can expect to the automated taxi agent to reach to the destination B.

Levels of KBAs

- Knowledge-based agents can be classified into three levels:

- ## Implementation Level
  - Implementation level is the **lowest level** of abstraction in a knowledge-based agent.
  - It **describes** how the knowledge and inference engine is implemented using a programming language.
  - The implementation level is **concerned** with the *details of the programming language* and the *algorithms used* to *implement* the *knowledge* and *inference engine*.

Actions Performed by the Agent

- **Inference System** is used when we want to **update** some **information** (sentences) in **Knowledge-Based System** and to **know** the **already present information**.
- **Mechanism is done** by **TELL** and **ASK** operations.
- **Inference include i.e.** ***Producing new sentences from old.***
  - **Inference** must **accept needs** when **one asks a question** to **KB** and **answer should follow** from what has been **Told** to **KB**.
  - **Agent also has a KB**, which initially has some **background Knowledge**.
  - Whenever, **agent** program **is called**, it **performs** some **actions**

Actions Performed by the Agent

- KBA engage in 3 primary operations to demonstrate intelligent behavior:
- **TELL:**
  - Agent **informs** the **knowledge base** about the **information** it has **perceived** from the **environment**.
  - This operation **allows the knowledge** base to be **continually updated** with **new facts**, ensuring the **agent's decisions** are based on the **most current information.**
- **ASK:**
  - Agent **queries** the **knowledge base** to **determine** the **best course** of **action** based on the available knowledge.
  - This **operation** is **crucial** for **decision-making**, allowing the agent to evaluate various options before taking action.

Actions Performed by the Agent

- **PERFORM:**
  - **Based** on the **knowledge base's recommendation**, the **agent** executes the **selected action**.
  - This **operation demonstrates** the **agent's ability** to **interact** with and **impact** its **environment effectively**.

Designing a KBA

- **Define the Domain and Scope**
  - **Domain Understanding:**
    - Clearly **define** the **domain** in which the **agent** will **operate**.
    - Understanding the domain **helps in identifying the type of knowledge** that needs to be **represented** and the **complexity of interactions** the agent will handle.
  - **Scope Definition:**
    - Determine the **scope of the agent's capabilities** and **functionalities**.
    - This includes **specifying the tasks it will perform** and the **decisions** it will make.
- **Choose the Right Knowledge Representation**
  - **Selecting Representation Techniques:**
    - The choice of **knowledge representation (KR)** technique is **critical**.
    - Common KR techniques include **semantic networks**, **frames**, **rules**, and **ontologies**.
    - Each has its strengths and is suited to different types of knowledge and reasoning processes.
  - **Representation Language:**
    - Choose a **suitable knowledge representation language** that can **express the complexity of the domain effectively**.
    - Languages such as **OWL** (Web Ontology Language), **RDF** (Resource Description Framework), and **rule-based languages** are popular choices.

Designing a KBA

- **Develop the Knowledge Base**
  - **Gathering Knowledge:**
    - **Collect comprehensive** and **accurate domain knowledge** from subject matter experts, literature, and existing databases.
    - This knowledge forms the foundation of the agent's decision-making capabilities.
  - **Knowledge Organization:**
    - **Organize** the **knowledge** in a **structured manner** that facilitates **efficient retrieval** and **reasoning**.
    - This includes categorizing knowledge, **defining relationships**, and **establishing hierarchies**.

- **Implement the Inference Engine**
  - **Reasoning Mechanisms:**
    - The **inference engine** should **employ reasoning mechanisms** suitable for the type of knowledge and the tasks at hand.
    - Deductive reasoning, inductive reasoning, and abduction are common approaches.
  - **Algorithm Selection:**
    - Choose **algorithms** that **optimize** the **reasoning process**, considering factors such as the complexity of queries, the size of the knowledge base, and the need for real-time responses.

Designing a KBA

- **Ensure Adaptability and Learning**
  - **Incorporating Learning:**
    - Design the agent with mechanisms to learn from new information and experiences.
    - Techniques such as machine learning algorithms can be integrated to update the knowledge base dynamically.
  - **Feedback Loops:**
    - Implement feedback loops that allow the agent to refine its knowledge and reasoning processes based on outcomes and external feedback.
- **Address Ethical and Security Considerations**
  - **Ethical Guidelines:**
    - Adhere to ethical guidelines in AI development to ensure that the agent's decisions and actions are fair, transparent, and respectful of privacy.
  - **Security Measures:**
    - Implement security measures to protect the knowledge base from unauthorized access and ensure the integrity of the information.
- **Test and Refine**
  - **Prototyping and Testing:**
    - Develop prototypes and conduct thorough testing to evaluate the agent's performance across various scenarios.
    - This helps in identifying gaps in the knowledge base and inefficiencies in the reasoning process.
  - **Iterative Refinement:**
    - Refine the agent iteratively based on testing outcomes, feedback from users, and evolving domain knowledge.

Challenges in Designing a KBA

- **Complexity of Knowledge Representation**
  - **Diverse and Complex Domains:**
    - Representing knowledge from complex domains accurately is a significant challenge.
    - Each domain may have its unique concepts, relationships, and rules that need to be captured in the knowledge base.
  - **Dynamic Knowledge:**
    - Keeping the knowledge base updated with the latest information and ensuring it adapts to changes in the domain or environment adds another layer of complexity.

- **Scalability and Performance**
  - **Handling Large Datasets:**
    - As the knowledge base grows, maintaining performance and scalability becomes challenging.
    - Efficient algorithms and data structures are required to ensure quick access and updates.
  - **Real-Time Processing:**
    - For applications that require real-time decision-making, optimizing the inference engine to process queries quickly without compromising accuracy is crucial.

Challenges in Designing a KBA

- **Ensuring Accuracy and Reliability**
  - **Data Quality:**
    - Effectiveness of a knowledge-based agent heavily depends on the **quality of the data** in its knowledge base.
    - Ensuring **accuracy**, **relevance**, and **completeness** of this data is a constant challenge.
  - **Error Handling:**
    - Developing **robust mechanisms** for **handling errors** or **inconsistencies** in the knowledge base is essential for maintaining the reliability of the agent's decisions and actions.

- **Ethical and Societal Considerations**
  - **Bias and Fairness:**
    - **Mitigating bias** in **knowledge-based** systems is a significant ethical concern.
    - Biases in the data or in the decision-making algorithms can lead to unfair or discriminatory outcomes.
  - **Transparency and Explainability:**
    - Ensuring that the agent's decision-making process is transparent and explainable is important for trust and accountability.
    - This is especially challenging with complex inference engines and large knowledge bases.

Propositional Logic

Propositional Logic

- A **proposition** is a **declarative statement** which is either **true** or **false**.
  - **Knowledge representation** in **logical** and **mathematical form**.
  - **Propositions**: Can be either **true** or **false**, but it **cannot** be **both**.
- **Propositional logic**: Consists of an ***object***, ***relations*** or ***function***, and ***logical connectives***.
- **Connectives:** can be said as a **logical operator** which **connects two sentences**.
- **Tautology**: **Proposition Formula** which is **always true**, and it is also called a **valid sentence**.
- **Contradiction**: A proposition formula which is **always false**.

a) It is Sunday.

b) The Sun rises from West (False proposition)

c) 3+3= 7(False proposition)

d) 5 is a prime number.

Propositional Logic

- **Propositional calculus / Propositional logic** : Area of logic which deals with **propositions**
  - Propositions are combined together using *Logical Connectives* or *Logical Operators*.
- **Syntax of Propositional Logic:**
  - Defines the *allowable sentences* for the *knowledge representation*.

- Two types of Propositions:

Atomic Propositions

- Atomic propositions are the **simple propositions**.
- It consists of a **single proposition symbol**.
- These are the ***sentences*** which must be either ***true*** or ***false***.

Compound propositions

- Are ***constructed*** by ***combining*** ***simpler*** or ***atomic*** ***propositions***, using parenthesis and logical connectives

(a) Atomic Propositions

a) 2+2 is 4, it is an atomic proposition as it is a **true** fact.

b) "The Sun is cold" is also a proposition as it is a **false** fact.

(b) Compound Propositions

a) "It is raining today, and street is wet."

b) "Ankit is a doctor, and his clinic is in Mumbai."

Propositional Logic

- **Propositional Logic – Logical Connectives**
  - Logical connectives are used to **connect two simpler propositions** or **representing a sentence logically**.
  - A **compound propositions** can be **created** with the **help** of **logical connectives**.
- **Five connectives:**
- **Negation**:
  - A sentence such as **¬ P** is called **negation of P**.
  - A literal can be either **Positive** literal or **negative literal**.
- **Conjunction**:
  - A sentence which has **∧** connective such as, **P ∧ Q** is called a **conjunction**.
  - ***Example***: Roshan is intelligent and hardworking. It can be written as,
  - P= Roshan is intelligent,
  - Q= Roshan is hardworking. → P∧ Q.
- **Disjunction**:
  - A sentence which has **∨** connective, such as **P ∨ Q**. is called **disjunction**.
  - Example: "Ritika is a doctor or Engineer",
  - Here P= Ritika is Doctor. Q= Ritika is Engineer, so we can write it as P ∨ Q.

Propositional Logic

- **Implication:**
  - A sentence such as **P → Q**, is called an **implication**.
  - Implications are also known as **if-then rules**.
  - ***Example***: If it is raining, then the street is wet.
  - Let P= It is raining, and Q= Street is wet, so it is represented as P → Q
- **Biconditional**:
  - A sentence such as **P⇔ Q** is a **Biconditional** sentence,
  - ***Example*** You will pass the exam if and only if you will work hard.
  - P= You will pass the exam, Q= you will work hard, it can be represented as P ⇔ Q.

Connective Symbols	Word	Technical Term	Example
$\wedge$	AND	Conjuction	$A \wedge B$
$\vee$	OR	Disjunction	$A \vee B$
$\rightarrow$	Implies	Implication	$A \rightarrow B$
$\Leftrightarrow$	If and only if	Biconditional	$A \Leftrightarrow B$
$\neg$ *or* $\sim$	NOT	Negation	$\neg A$ *or* $\neg B$

Propositional Logic

- Propositional Logic – Truth Table

<table>
<tr><th>P</th><th>Q</th><th>$\neg P$
Negation</th><th>$P \wedge Q$
Conjunction</th><th>$P \vee Q$
Disjunction</th><th>$P \rightarrow Q$
Implication</th><th>$P \Leftrightarrow Q$
Biconditional</th></tr>
<tr><td>True</td><td>True</td><td rowspan="2">False</td><td>True</td><td>True</td><td>True</td><td>True</td></tr>
<tr><td>True</td><td>False</td><td>False</td><td>True</td><td>True</td><td>False</td></tr>
<tr><td>False</td><td>True</td><td rowspan="2">True</td><td>False</td><td>True</td><td>False</td><td>False</td></tr>
<tr><td>False</td><td>False</td><td>False</td><td>False</td><td>True</td><td>True</td></tr>
</table>

Propositional Logic

- Propositional Logic – Truth Table

Precedence	Operators
1st Precedence	Parenthesis
2nd Precedence	Negation
3rd Precedence	Conjunction (AND)
4th Precedence	Disjunction (OR)
5th Precedence	Implication
6th Precedence	Biconditional

In $\neg \boldsymbol{R} \vee \boldsymbol{Q}$ can be written as $(\neg \boldsymbol{R}) \vee \boldsymbol{Q}$

Propositional Logic

- Properties of operators

- **Commutativity:**
  - $P \wedge Q = Q \wedge P$, or
  - $P \vee Q = Q \vee P$.
- **Associativity:**
  - $(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$,
  - $(P \vee Q) \vee R = P \vee (Q \vee R)$
- **Identity element:**
  - $P \wedge True = P$,
  - $P \vee True = True$.
- **Distributive:**
  - $P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$.
  - $P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$.
- **DE Morgan's Law:**
  - $\neg (P \wedge Q) = (\neg P) \vee (\neg Q)$
  - $\neg (P \vee Q) = (\neg P) \wedge (\neg Q)$.
- **Double-negation elimination:**
  - $\neg (\neg P) = P$.

Components of a Propositional Logic Agent

- **Knowledge Base (KB):**
  - A **collection** of **propositional formulas** (or statements) that represent the knowledge the agent has about the world. These formulas are typically written using **logical connectives**.
  - For example,
    - if the agent knows that "It is raining" (R) and "If it is raining, then the ground is wet" ($R \rightarrow W$),
    - The knowledge base would include these propositions.

- **Inference Mechanism:**
  - **Allows** the **agent** to **derive new information** from the **knowledge base**.
  - Common inference methods include:
    - **Modus Ponens**: If we know "**$P \rightarrow Q$**" and "**P**," **we can infer** "**Q**."
    - **Resolution**: A method used in automated theorem proving to infer conclusions from the knowledge base.
  - Inference mechanism **helps** the **agent make logical conclusions** and **update** its **knowledge base** accordingly.

Components of a Propositional Logic Agent

- **Query Answering:**
  - When **an agent needs to determine** if a certain **proposition** (e.g., "The ground is wet") is **true**, it uses the inference mechanism to check if this proposition can be derived from the knowledge base.
  - ***For example***,
    - **Given** the **knowledge base** with "$R \rightarrow W$" and the fact "R" (***It is raining***), the agent can infer "**W**" (The ground is **wet**).

Components of a Propositional Logic Agent

- Consider an agent with the following knowledge base:
- ***Propositions:***
  - **P**: The door is open.
  - **Q**: The alarm is ringing.
  - **R**: The window is broken.
- ***Rules:***
  - **Rule 1**: If the **door is open** and the **alarm is not ringing**, then the **window is broken**. ($P \land \neg Q \rightarrow R$)
  - **Rule 2**: If the **alarm is ringing**, then the **door is open**. ($Q \rightarrow P$)
- ***Initial Knowledge:***
  - The **door** is **open** (P).
  - The **alarm** is **ringing** (Q).
- ***Inference:***
  - From **Rule 2** and the **initial knowledge** (Q and P), we **confirm** the **door is open** (P) is consistent.
  - From **Rule 1** and the **knowledge** (P and Q), we know the **window must be broken** (R) because the **condition** $P \land \neg Q \rightarrow R$ is met.

Propositional Logic Agent

- Advantages and Limitations

- **Advantages:**
  - **Simplicity**: Propositional logic is **relatively straightforward**, making it easy to understand and implement.
  - **Decidability**: Propositional logic is **decidable**, meaning there is an algorithm that can determine whether any **given proposition** is **true** or **false** based on the **knowledge base**.

- **Limitations:**
  - **Expressiveness**: Propositional logic is **limited** in its **ability** to **express more complex statements** about the **world**, especially when dealing with statements involving variables or quantifiers (which require predicate logic).
  - **Scalability**: As the **number** of **propositions grows**, the **size** of the **knowledge base** and the **complexity** of **inference** can become **unwieldy**.

Propositional Logic Agent

- **Applications**

- **Automated Theorem Proving**: Propositional logic is used to prove theorems by checking if they can be logically derived from a set of axioms.
- **Planning**: In planning problems, agents use propositional logic to generate a sequence of actions that achieve a desired goal.
- **Diagnosis**: Agents use propositional logic to identify the cause of problems based on symptoms and known relationships.
- **Knowledge Representation**: Used in expert systems to encode domain-specific knowledge, enabling systems to make informed decisions or provide recommendations.
- **Automated Reasoning**: Employed in legal reasoning systems to evaluate legal arguments or in configuration systems to ensure that all parts of a product configuration are compatible.
- **Game Playing and Strategy**: Used in simple board games or puzzle-solving applications where the environment and rules are well-defined.
- **Intelligent Tutoring Systems**: Used in educational technology to provide personalized feedback and guidance based on a student’s responses.

Propositional Logic Agent

Q1: Consider these statements,

P: It will rain today.

Q: I shall go to the party.

Solve these propositions with reference to the above statements:

(i) (~P V~Q)

(ii) (~P ∧ Q)

(iii) (P V Q)

Solution:

(i) (~P V~Q) : It will not rain today or I shall not go to the party.

(ii) (~P ∧ Q) : It will not rain today and I shall go to the party.

(iii) (P V Q) : It will rain today or I shall go to the party.

First-order Logic /
Predicate Logic /
First-order Predicate Logic

First Order Logic (FOL)

- **Propositional Logic (PL)**, which can *only represent the facts*, which are either **true** or **false**.
- **PL** is **not sufficient** to **represent** the **complex sentences** or natural language statements.
- PL has very **limited expressive power**.
- *Consider the following sentence, which we cannot represent using PL logic.*
  - "Some humans are intelligent", or
  - "Sachin likes cricket."

- **First-order logic** is another way of knowledge representation: An **extension to PL**.
- FOL is **sufficiently expressive** to represent the ***natural language statements*** in a ***concise way***.
- FOL 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 : *Components*

- **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:

- **Objects**: A, B, people, numbers, colors, theories, squares, …
- **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, …
- **Function**: Father of, best friend, third inning of, end of, …
- **Constants**: ***Constants*** are ***symbols*** that ***represent specific objects*** in the ***domain***.
  - ***Examples***: *If a, b, and c are constants, they might represent specific individuals like Alice, Bob, and Charlie.*
- **Variables**: Variables are **symbols** that can **represent any object** in the **domain**.
  - ***Examples***: *Variables such as x, y, and z can represent any object in the domain.*
- **Predicates**: Predicates represent ***properties*** of objects or ***relationships between objects***.
  - ***Examples***: *P(x) could mean "x is a person", while Q(x, y) could mean "x is friends with y".*
- **Functions**: Functions **map objects** to **other objects**.
  - ***Examples***: *f(x) could represent a function that maps an object x to another object, like "the father of x".*

First Order Logic : Components

- **Logical Connectives:**
  - These include ∧ (and), ∨ (or), ¬ (not), → (implies), and ↔ (if and only if).
  - *Examples*: *P(x) ∧ Q(x, y)* means "*P(x) and Q(x, y)* are both *true*".
- **Equality:**
  - States that two objects are the same.
  - Examples: x = y asserts that x and y refer to the same object.

Syntax of First-Order Logic

The syntax of **FOL** defines the rules for constructing well-formed formulas:

- **Atomic Formulas**: The simplest formulas, which can be predicates applied to terms
  - e.g., $P(a), Q(x, y)$
- **Complex Formulas**: Formed by combining atomic formulas using **logical connectives** and **quantifiers.**
  - e.g., $\forall \boldsymbol{x}\ (\boldsymbol{P}(\boldsymbol{x}) \vee \neg \boldsymbol{Q}(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{y})))$

First Order Logic : Components

Semantics of First-Order Logic

The semantics define the meaning of FOL statements:

- **Domain**: A **non-empty set** of **objects** over which the variables range.

- **Interpretation**: ***Assigns meanings*** to the ***constants***, ***functions***, and ***predicates***,
  - Specifying ***which objects*** the ***constants refer*** to,
  - Which ***function*** the ***function symbols denote***, and
  - Which ***relations*** the ***predicate symbols denote***.

- **Truth Assignment**: Determines the ***truth value*** of ***each formula*** based on the interpretation.

First Order Logic : Key Components

- Two main components:
- **Syntax**:
- Syntax represents the **rules to write expressions** in First Order Logic in Artificial Intelligence.

- **Semantics**:
- Semantics refers to the **techniques** that **we use to evaluate an expression** of First Order Logic in AI.
- These techniques use various known ***relations*** and ***facts*** of the ***respective environment*** to deduce the ***boolean value*** of the ***given*** First Order Logic ***expression***.

Element	Example	Meaning
Constant	1, 2, A, John, Mumbai, cat, ...	Values that can not be changed
Variables	x, y, z, a, b, ...	Can take up any value and can also change
Predicates	Brother, Father, >, ...	Defines a relationship between its input terms
Function	sqrt, LeftLegOf, ...	Computes a defined relation of input term
Connectives	$\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$	Used to form complex sentences using atomic sentences
Equality	==	Relational operator that checks equality
Quantifier	$\forall$, $\exists$	Imposes a quantity on the respective variable

First Order Logic : Key Components

- **Atomic Sentences**:
  - Atomic sentences are the **most basic expressions** of First Order Logic in AI.
  - These sentences comprise a ***predicate*** followed by a ***set*** of ***terms inside*** a ***parenthesis***.
  - *Formally stating, the structure of an atomic sentence looks like the following.*
    - Predicate 1 (term 1, term 2, term 3,...)
    - Predicate 2 (term 2, term 4,...)

First Order Logic : Key Components

- **Complex Sentences:**
  - Complex sentences can be **constructed** by **combining atomic sentences** using **connectives** like
  - AND (∧), OR (∨), NOT (¬), IMPLIES (⇒), IF AND ONLY IF (⇔) etc.

  - Formally stating, if c_1, c_2, ... c_3,c_4... represent connectives, a complex sentence in First Order Logic in AI can be defined as follows.
  - Predicate 1(term 1, term 2,...) **c1** Predicate 2(term 1, term 2,...)

  - For example, in the expression, **Polygon (Rectangle)**, which is a translation of "***Rectangle is a polygon***." in First Order Logic in Artificial Intelligence,
  - "Rectangle" is the subject, and "Polygon" represents a predicate.

First Order Logic : Key Components

- **Quantifiers in First-Order Logic: Quantify** any **entity** in a **given environment**.
  - Quantification refers to the *identification of the **total number** of an entity* that is *present in the environment* and *satisfies a given expression* in First Order Logic in Artificial Intelligence.
  - Quantifiers enable us to determine the range and scope of a variable in a logical expression.
- ***Two types of quantifiers :***
  - Universal &
  - Existential Quantifier

First Order Logic : Key Components

- **Universal Quantifier:**
  - Is a ***symbol*** in a ***logical expression*** that signifies that the ***given expression is true in its range for all instances*** of the concerned entity.
  - **Represented** by the **symbol ∀** (an inverted A).
  - If *x* is a variable, then ∀x is read as "For all x" or "For every x" or "For each x".

- ***For example:***
  - Let us take the sentence, "*All cats like fish*".
  - Let us take a **variable x** which can take the **value of "cat".**
  - Let us take a **predicate cat (x)** which is **true** if **x is a cat**.
  - Similarly, let us take another **predicate likes (x, y)** which is ***true*** if ***x likes y***.
  - Therefore, using the ***universal quantifier ∀***, we can write

$$\forall x, cat\ (x) \Rightarrow\ likes\ (x, fish\ ).$$

  - This expression is read as "*For all x, if x is a cat, then x likes to fish*".

First Order Logic : Key Components

- **Existential Quantifier**
  - Is a symbol in a logical expression that signifies that the ***given expression*** is ***true*** in ***its range*** for ***at least one*** of ***the instances*** of the concerned entity.
  - It is represented by the **symbol ∃** (an inverted E).
  - If *x is a variable*, then *∃x* is read as "*There exists x*" or "*For some x*" or "*For at least one x*".

  - ***For example:*** "*Some students like ice cream*".
  - Let us take a **variable x** which can take the **value of "student".**
  - Let us take a predicate student (x), which is true if x is a student.

  - Similarly, let us take another predicate *likes (x,y),* which is *true if xx likes yy*.
  - Therefore, using the existential quantifier ∃, we can write
    - $\exists\, \boldsymbol{x\ student}\ (\boldsymbol{x})\ \wedge\ \boldsymbol{likes}\ (\boldsymbol{x}, \boldsymbol{ice} - \boldsymbol{cream}\ )$
  - This expression reads, "*There exists some x such that x is a student and also likes ice cream*".

First Order Logic : Key Components

- Examples of FOL using quantifier:

- ***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.

$$\forall x\ \boldsymbol{bird}(\boldsymbol{x}) \rightarrow \boldsymbol{fly}(\boldsymbol{x})$$

- ***Everyone respects their parents.***
  In this question, the predicate is "**respect(x, y)**," where **x=man/women**, and **y= parent**.
  Since there is every one so will use ∀, and it will be represented as follows:

$$\forall \boldsymbol{x}\ \boldsymbol{Everyone}\ (\boldsymbol{x}) \rightarrow \boldsymbol{respects}\ (\boldsymbol{x}, \boldsymbol{parent})$$

First Order Logic : Key Components

- Examples of FOL using quantifier:
- ***Some boys play cricket.***
  In this question, 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:
  $$\exists x\ \boldsymbol{boys(x)} \rightarrow \boldsymbol{play(x, cricket)}$$
- ***Not all students like both Mathematics and Science.***
  In this question, the predicate is "**like(x, y)**," where *x= student*, and *y= subject*.
  Since there are ***not all students***, so we will use ***∀ with negation***, so following representation for this:
  $$\neg\forall\ (x)\ [\ \boldsymbol{student(x)} \rightarrow \boldsymbol{like(x, Mathematics)} \land \boldsymbol{like(x, Science)}]$$

First Order Logic : Key Components

- While using quantifiers in writing expressions for FOL, we need to keep the following points in mind.
- **Main connective** for the **universal quantifier ∀** is the **implication (⟹).**
- **Main connective** for **existential quantifier ∃** is and **(∧)**.

- **Types of Variables:** Free and Bound Variables
  - Based upon their interaction with the quantifiers.

- **Free Variables:**
  - Those variables that ***do not come under the scope of the quantifier***.
  - For instance, in an expression $\forall \boldsymbol{x} \exists \boldsymbol{y} \boldsymbol{P}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$, z is a free variable because it doesn't come under the scope of any quantifier.

- **Bound Variables:**
  - Those variables that ***occur inside the scope of the quantifier***.
  - For instance, in an expression $\forall x \exists y P(x, y, z)$, x and y are bound variables because they occur inside the scope of the quantifiers.

First Order Logic : Challenges

- Complexity:
  - Representing certain real-world domains accurately in FOL can lead *to complex and unwieldy formulas*, making reasoning and inference computationally expensive.
- Expressiveness Limitations:
  - FOL has *limitations* in representing *uncertainty*, *vagueness*, and *probabilistic relationships*, which are common in many AI applications.
- Knowledge Acquisition:
  - *Encoding knowledge* into FOL *requires expertise* and *manual effort*, making it challenging to scale and maintain large knowledge bases.
- Inference Scalability:
  - *Reasoning* in FOL can be *computationally intensive*, especially in *large knowledge bases*, requiring efficient inference algorithms and optimization techniques.
- Handling Incomplete Information:
  - FOL *struggles* with *representing* and *reasoning* with *incomplete* or *uncertain information*, which is common in real-world applications.

First Order Logic : Limitations

- Inability to Represent Recursive Structures:
  - FOL cannot directly represent recursive structures, limiting its ability to model certain types of relationships and processes.
- Lack of Higher-Order Reasoning:
  - FOL lacks support for higher-order logic, preventing it from representing and reasoning about properties of predicates or functions.
- Difficulty in Representing Context and Dynamics:
  - FOL struggles with representing dynamic or context-dependent knowledge, such as *temporal relationships* or *changes over time*.
- Limited Representation of Non-binary Relations:
  - FOL *primarily deals with binary relations*, making it *less suitable* for *representing complex relationships* involving *multiple entities*.
- Difficulty in Handling Non-monotonic Reasoning:
  - FOL is *not well-suited* for *non-monotonic reasoning*, where *new information* can lead to *retraction* or *modification* of *previously inferred conclusions*.