Name: Ritik Kumar Sriavstava

Admission number:18SCSE1180051

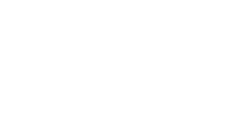
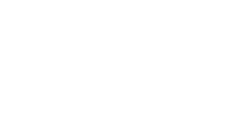
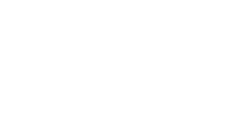
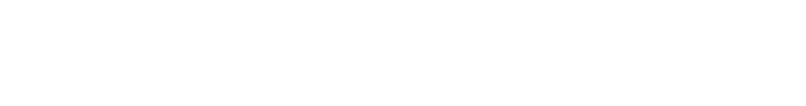
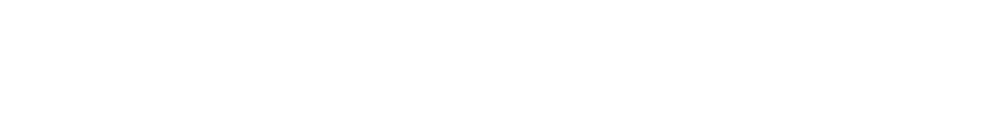
**Introduction**  
Consider the following example. We need to convert the following sentence into a mathematical statement using propositional logic only.

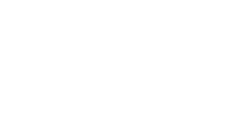
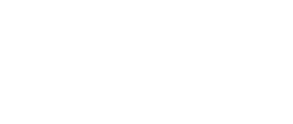
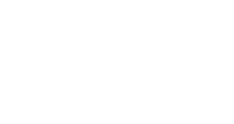
"Every person who is 18 years or older, is eligible to vote."

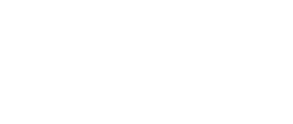
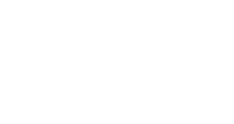
The above statement cannot be adequately expressed using only propositional logic. The problem in trying to do so is that propositional logic is not expressive enough to deal with quantified variables. It would have been easier if the statement were referring to a specific person. But since it is not the case and the statement applies to all people who are 18 years or older, we are stuck.  
Therefore we need a more powerful type of logic.

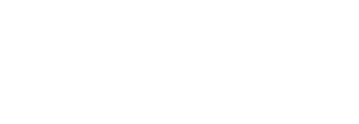
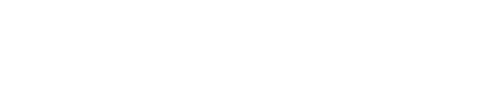
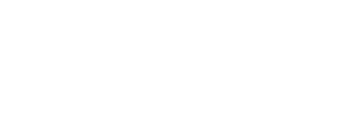
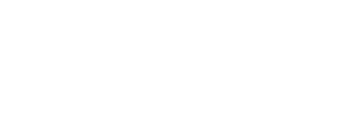
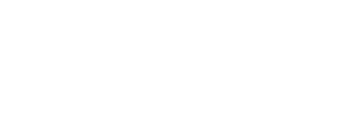
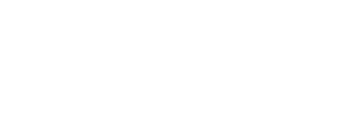
**Predicate Logic**  
Predicate logic is an extension of Propositional logic. It adds the concept of predicates and quantifiers to better capture the meaning of statements that cannot be adequately expressed by propositional logic.

**What is a predicate?**

Consider the statement, “ is greater than 3″. It has two parts. The first part, the variable , is the subject of the statement. The second part, “is greater than 3”, is the **predicate**. It refers to a property that the subject of the statement can have.  
The statement “ is greater than 3″ can be denoted by  where  denotes the predicate “is greater than 3” and  is the variable.  
The predicate  can be considered as a function. It tells the truth value of the statement  at . Once a value has been assigned to the variable , the statement  becomes a proposition and has a truth or false(tf) value.  
In general, a statement involving n variables  can be denoted by . Here  is also referred to as **n-place predicate** or a **n-ary predicate**.

* **Example 1:** Let  denote the statement “ > 10″. What are the truth values of  and ?

**Solution:**  is equivalent to the statement 11 > 10, which is True.  
 is equivalent to the statement 5 > 10, which is False.

* **Example 2:** Let  denote the statement ““.  What is the truth value of the propositions  and ?  
  **Solution:**  is the statement 1 = 3 + 1, which is False.  
   is the statement 2 = 1 + 1, which is True.

In [mathematical logic](https://en.wikipedia.org/wiki/Mathematical_logic), a **predicate** is commonly understood to be a [Boolean-valued function](https://en.wikipedia.org/wiki/Boolean-valued_function) *P*: *X*→ {true, false}, called a predicate on *X*. However, predicates have many different uses and interpretations in mathematics and logic, and their precise definition, meaning and use will vary from theory to theory. So, for example, when a theory defines the concept of a [relation](https://en.wikipedia.org/wiki/Relation_(mathematics)), then a predicate is simply the [characteristic function](https://en.wikipedia.org/wiki/Indicator_function) (otherwise known as the [indicator function](https://en.wikipedia.org/wiki/Indicator_function)) of a relation. However, not all theories have relations, or are founded on [set theory](https://en.wikipedia.org/wiki/Set_theory), and so one must be careful with the proper definition and semantic interpretation of a predicate.

Simplified overview[[edit](https://en.wikipedia.org/w/index.php?title=Predicate_(mathematical_logic)&action=edit&section=1)]

Informally, a predicate is a statement that may be true or false depending on the values of its variables.[[1]](https://en.wikipedia.org/wiki/Predicate_(mathematical_logic)#cite_note-1) It can be thought of as an operator or function that returns a value that is either true or false.[[2]](https://en.wikipedia.org/wiki/Predicate_(mathematical_logic)#cite_note-2) For example, predicates are sometimes used to indicate set membership: when talking about sets, it is sometimes inconvenient or impossible to describe a set by listing all of its elements. Thus, a predicate *P(x)* will be true or false, depending on whether *x* belongs to a set.

Predicates are also commonly used to talk about the [properties](https://en.wikipedia.org/wiki/Property_(philosophy)) of objects, by defining the set of all objects that have some property in common. So, for example, when *P* is a predicate on *X*, one might sometimes say *P* is a [property](https://en.wikipedia.org/wiki/Property_(philosophy)) of *X*. Similarly, the notation *P*(*x*) is used to denote a sentence or statement *P* concerning the variable object x. The set defined by *P*(*x*) is written as {*x* | *P*(*x*)}, and is the set of objects for which *P* is true.

For instance, {*x* | *x* is a natural number less than 4} is the set {1,2,3}.

If *t* is an element of the set {*x* | *P*(*x*)}, then the statement *P*(*t*) is *true*.

Here, *P*(*x*) is referred to as the *predicate*, and *x* the *placeholder* of the [*proposition*](https://en.wikipedia.org/wiki/Proposition). Sometimes, *P*(*x*) is also called a ([template](https://en.wikipedia.org/wiki/Template_processor) in the role of) [propositional function](https://en.wikipedia.org/wiki/Propositional_function), as each choice of the placeholder *x* produces a proposition.

A simple form of predicate is a [Boolean expression](https://en.wikipedia.org/wiki/Boolean_expression), in which case the inputs to the expression are themselves Boolean values, combined using Boolean operations. Similarly, a Boolean expression with inputs predicates is itself a more complex predicate.

[**next →**](https://www.javatpoint.com/ai-knowledge-engineering-in-first-order-logic)[**← prev**](https://www.javatpoint.com/ai-knowledge-base-for-wumpus-world)

# **First-Order Logic in Artificial intelligence**

In the topic of Propositional logic, we have seen that how to 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:

* 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:
  + **Objects:** A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ......
  + **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, ......
* As a natural language, first-order logic also has two main parts:
  1. **Syntax**
  2. **Semantics**

**Syntax of first order**

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

### **Basic Elements of First-order logic:**

* Following are the basic elements of FOL syntax:

|  |  |
| --- | --- |
| **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)**.

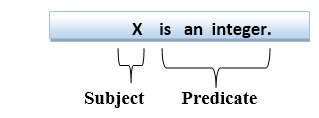
### **Complex Sentences:**

* 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:

* A quantifier is a language element which generates quantification, and quantification specifies the quantity of specimen in the universe of discourse.
* These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression. There are two types of quantifier:
  1. **Universal Quantifier, (for all, everyone, everything)**
  2. **Existential quantifier, (for some, at least one).**