# Module 1 Lecture Notes

## Introduction to Mathematical Logic

Mathematical logic attempts to formalize how we reason. It evolved as a field since the 19th and the 20th centuries, when mathematicians looked for ways to provide strong logical foundations for mathematics. Eventually it became also part of the foundations of Computer Science, serving both a theoretical and a practical purpose. When a person learns a programming language for the first time, he/she will be exposed to principles of mathematic logic, for example, when studying decision statements such as the "IF" statement:

```
if ((x>=0) && (x<=100)) System.out.println(x + " is in the [0, 100] range").
```

You will learn in this course about two parts of mathematical logic: the *propositional logic* (also called *propositional calculus*) and the *predicate logic* (also called *predicate calculus* or *first-order logic*). Mathematical logic does not end with these two parts, there are higher order logics too. Another area you will explore will be logic programming, in particular, basic aspects of the Prolog language.

Propositional logic deals with the construction of complex statements, obtained by combining simpler ones with logical connections such as *and* or *or*. It also involves the study of the properties that are derived by the ways we construct those complex propositions. Consider for instance the two propositions *A* and *B*:

*A: I have time*
*B: I have money*

From what you have learned in the programming courses, you can see that the truth of the more complex proposition *A and B* depends on the truth of both *A* and *B*. This is the type of relationship that propositional logic studies: how the truth of more complex statements is obtained from simpler ones.

First-order logic extends propositional logic and include other concepts not covered by it, such as variables, functions, relations, and quantifiers. Here you have an example of a mathematical definition of a well-known object in the subject of data structures and algorithms, the Big-Oh notation:

| |
|---|
| *Definition* |
| Let f and g be functions, $f:\mathbb{R}\to\mathbb{R}$, $g:\mathbb{R}\to\mathbb{R}$. We say that f is Big-Oh of g (denoted by f = O(g)): |
| if there are positive constants c and $n_0$ such that $f(x) \le c\, g(x)$ for $x > n_0$. |

This definition can be expressed in a more formal way using the language of first-order logic:

*Definition*

Let f and g be functions, $f:\mathbb{R}\rightarrow\mathbb{R}$, $g:\mathbb{R}\rightarrow\mathbb{R}$. We say that f is Big-Oh of g (denoted by f = O(g)):

$\exists c>0\ \exists n_0>0\ \forall x>n_0\ (f(x) \leq c\ g(x))$

which involves not only variables, functions, and relations (≤, >) but also quantifiers (∃, ∀).

## Application of Mathematical Logic to Computer Science

Logic is an essential part of the theoretical foundations of Computer Science; it has been called the calculus of computer science. It is used in subjects such as reasoning automation in artificial intelligence, in finite automata theory, in software specification and verification, and in automatic theorem proving.

## Review on Sets

A fundamental concept in our course is that of a mathematical set. Remember that a set is defined as a collection of distinct objects; note that the objects must be unique in sets, and if duplicates are to be allowed, then we use the term *multiset*. A special set is the *empty set*, the set that contains no elements which is denoted by ∅. If P is a set and Q is a subset of P, denoted by $Q \subseteq P$, then $Q^C = P - Q$ is called the *complement* of Q in P.

Basic operations associated to sets include:

*Membership in a set*

This operation shows whether a given element is part of a set or not. In mathematics, we write $x \in S$ to indicate that x belongs to the set S and $x \notin S$ to indicate the opposite; for example, $5 \in \{1, 3, 5, 7\}$ and $4 \notin \{1, 3, 5, 7\}$.

*Union of two sets*

This operation would be the equivalent to addition of sets, and it is denoted by the symbol ∪. Let P and Q be sets, $P \cup Q$ is defined as the set consisting of elements either in S or R. For example, suppose that $P = \{1, 3, 5, 7\}$ and $Q = \{2, 5, 6, 7\}$, then $P \cup Q = \{1, 2, 3, 5, 6, 7\}$. Again, note that duplications are not allowed in sets, so if an element is a member of both P and Q (such as 5 and 7 in the example), it is listed only once in $P \cup Q$.

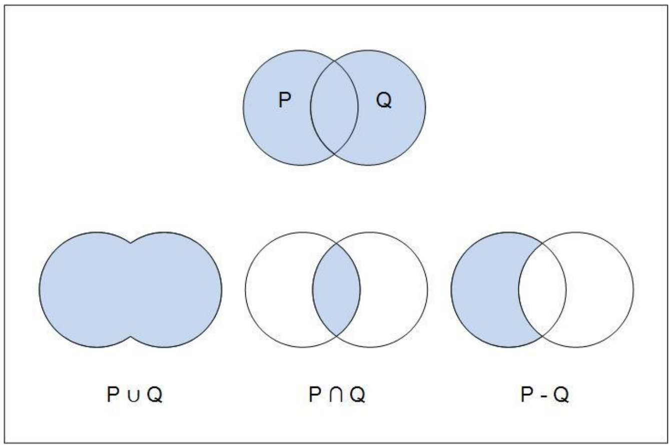
*Intersection of two sets*

The intersection of two sets P and Q, denoted by $P \cap Q$, is defined as the set whose members are members of both P and Q. For instance, if $P = \{1, 3, 5, 7\}$ and $Q = \{2, 5, 6, 7\}$, $P \cap Q = \{5, 7\}$.

*Difference of two sets*

The difference of two sets P and Q, denoted by P – Q, is defined as the set whose members are members of P but not members of Q. For instance, if P = {1, 3, 5, 7} and Q = {2, 5, 6, 7}, P - Q = {1, 3}.

These operations are illustrated using Venn diagrams below:




Set operations satisfy several identities (also called "laws"), including the identity, the complement, the idempotent, the commutative, the associative, and the distributive laws. You can see in the Appendix section of the textbook a discussion on these and other results related to sets.