Chapter 1

Introduction and Motivation

1.1 Discrete Mathematics and Computer Science

Discrete Mathematics is concerned with finite and countably infinite mathematical structures. There are at least three major reasons why discrete mathematics is of central importance in computer science

- **Discrete structures**
  Many objects studied in computer science are discrete mathematical objects, for example a graph modeling a computer network or an algebraic group used in cryptography. Many applications exploit sophisticated properties of the involved structures
- **Abstraction**
  A computer system can only be understood by considering a number of layers of abstraction, from application programs to the physical hardware
- **Programs as mathematicals objects**
  Understanding a computer program means to understand it as a discrete mathematical object

1.2 Discrete Mathematics: A Selection of Teasers

This section contains 8 examples, worth taking a look at.

1.3 Abstraction: Simplicity and Generality

In Discrete Mathematics, abstraction stands for simplicity and generality. In order to manage complexity, software systems are divided into components (called modules, layers, objects or abstract data types) that interact with each other and with the environment in a well-defined manner.

Abstraction means *simplification*. By an abstraction one ignores all aspects of a system that are not relevant for the problem at hand, concentrating on the properties that matter

Abstraction also means *generalization*. If one proves a property of a system described at an abstract level, then this property holds for any system with the same abstraction, independently of any details.

1.4 Programs as Discrete Mathematical Objects

A computer program is a discrete mathematical object. The correctness of a program is a (discrete) mathematical statement. The proof that a program is correct (i.e., that it satisfies its specification) is a mathematical proof, not a heuristic argument.

Chapter 2

Mathematical Reasoning, Proofs, and a First Approach to Logic

2.1 What is a Proof?

2.1.2 The concept of a proof

Definition 2.1 (Informal)
A *proof* of a statement S is a sequence of simple, easy verifiable consecutive steps. The proof starts from a set of axioms (things postulated to be true) and known (previously proved) facts. Each step corresponds to the application of a derivation rule to a few already proven statements, resulting in a newly proved statement, until the final step proves S.

2.1.3 Informal vs. Formal Proofs

Formal Proof: Uses mathematical symbols and language.
Informal Proof: Explained using common, everyday language.

There are at least three (related) reasons for using a more rigurous and formal type of proof:

- **Prevention of errors**
  A completely formal proof leaves no room for interpretation, and hence allows to exclude errors
- **Proof complexity and automatic verification**
  Certain proofs are too complex to be carried out and verified "by hand". A computer is required for the verification, which can only deal with rigurously formalized statements, not with semi-precise common language.
- **Precision and deeper understanding**
  A formal proof requires the formalization of the arguments and can lead to a deeper understanding (also for the author of the proof).

2.1.4 The Role of Logic