序

计算机的程序设计：

- 不关心程序具体的用途
- 关心性能
- 关心能不能和其他程序平滑衔接成更大的程序

程序员应该追求好的算法和惯用法。即使某些程序难以精确描述，程序员也要估计性能并且设法优化。

前言

本书的目标：我们要有好的程序设计风格要素和审美观：

- 程序必须写得好读
- 最基本的材料是用于控制大型软件系统的复杂性的
    - 包括：抽象、混合与匹配

本书的思想方法：过程性认识论

即如何从命令式的观点去研究知识的结构

即弄清楚要计算什么，怎么将问题分解为可控的部分，如何对可控的部分展开工作

- 所以Lisp很适合用来教学

从更多角度观察和理解程序和程序设计中的问题

- 函数式程序设计
- 各种程序组织方式，分解和控制程序复杂性的技术
- 丰富的编程模式
- 对一些基础问题的理解
- 上述各方面与常规（命令式，eg.C/C++等）程序设计的关系和启示等

说明式知识vs过程性知识

是什么vs怎么做

软件开发的工作就在说明式雨过程式知识的结合点上

命令式计算和函数式计算

1. 程序设计的主流：命令式程序设计。基于一组基本操作，在一个环境里运行，操作效果是改变环境的状态，体现在所创建和修改的状态中。
2. 函数式程序设计：计算过程被看成是数据的交换，程序的行为就是数据的一系列变换。

常规程序以命令方式描述计算，接近实际硬件，可能高效，但编程时需要关注很多细节

函数式程序设计层次较高，更抽象，程序可能更清晰，但需要复杂的运行时支持，可能效率较低

Lisp是函数式语言之祖

Scheme不是纯函数式语言，为了效率和对计算的控制，提供了命令式特征（状态操作）