# 100年后的编程语言——读书笔记

**蔡志杰 22112020002**

## 一、关于编程语言的一些基础描述

编程语言的两大组成部分：

1. 基本运算符的集合（公理）
2. 运算符以外的其他部分（可用基本运算符表示）

编程语言长期存在的最重要因素就是对基本运算符及其规模的选择，内核越小越干净的编程语言生命力越强。编程语言进化速度相比于软件发展要慢得多，因为其只是一种严格符合规则的描述，是一种缓慢的渐变式变化。

## 二、百年后的一些环境背景

百年后的计算机运行速度会快很多，更快的算力意味着编程语言需要应付更多的极端情况，涵盖更大范围的效率要求。同时在面对更快算力的时候，需要思考如何充分利用硬件更强大的性能，比如说在设计编程语言的时候在性能和便利性之间进行权衡。

## 三、百年后编程语言的特点

#### 3.1语言内核

语言内核设计的一些建议：首先需要将语言的语义与语言的实现予以分离，在语义上不设置冗余的公理，做到简洁干净（体现在数据结构的扁平化，去掉可被替代的数据结构，如字符串等）；而在实现上做好编译器的优化，使其内对简洁语义有针对性地优化。像以加快运行速度为目的使得编程语言语义大大复杂的行为不推荐，属于过早优化。此外对代码实现方式减少限制会使得程序具备更大的灵活性。

未来的编程语言应该是让你写程序最省力的语言，这里的“省力”还需要跳出当前编程语言的思维定势。解析树的大小往往可以用来衡量开发程序的难易度。最省力的语言同时可以理解成是到达目标的最短路径，不走弯路，设计编程语言的时候也应该这样。

#### 3.2语言种类

百年后的编程语言，编程语言会在运行速度和编程便利性之间进行取舍，做出不同取舍的编程语言都会存在。由于代码效率和机器性能的差距在不断增大，为了更好地提高或是衡量代码的效率，性能分析器显得尤为重要，好的性能分析器能帮助和指导程序的开发。

特定领域的小众语言将来会被设计成很薄的一层，便于看出其底层的通用型语言的实现，进而可以减少学习时间，降低使用成本。

#### 3.3语言的开发者

未来的语言更可能是开源语言，而非以研究性项目的形式出现，这与科学领域的研究现状相关，研究者可以做的题目往往与对设计优秀编程语言有帮助的题目交集很小。同时新语言的设计整将更多是本身就需要使用他们的程序员而非开发编译器研究者，这将有利于编程语言的发展。

## 四、其他部分

#### 4.1自下而上的思路

多层形式的开发（这里的多层指在硬件和应用软件中设置很多的软件层）虽然会消耗硬件性能，但是可用带来更小更灵活的程序，同时增加软件的可重用性。面向对象编程中自下而上的开发方法（而非面向对象编程本身）则与多层形式开发有异曲同工之处。

#### 4.2并行计算

通常对程序并行的优化是在编程后期才需要做的事情，百年后仍不会出现大量使用大规模并行计算的情况，否则也属于过早优化了。