Atitit atitit 编程语言之道补充 s22 attilax 艾龙 著

[1. 程序设计是个什么概念呢？历史发展 1](#_Toc16331)

[1.1. 连接电缆 2](#_Toc8974)

[1.2. 程序内置纸带打点 2](#_Toc21316)

[1.3. FORTRAN 语言问世 2](#_Toc19475)

[2. 懒惰：程序员的三大美德之一 2](#_Toc16126)

[3. 第 3 章　语法的诞生 2](#_Toc8486)

[3.2. 3.2　栈机器和 FORTH 语言 3](#_Toc4390)

[3.3. 3.3　语法树和 LISP 语言 3](#_Toc26672)

[3.4. 3.4　中缀表达式 前缀 后置表达式 3](#_Toc28328)

[3.5. S表达式 3](#_Toc30315)

[4. 第 5 章　函数 3](#_Toc5071)

[5. 错误处理 3](#_Toc19212)

[5.1. 返回值 vs 异常 3](#_Toc11976)

[5.2. 可以追加错误类型和自著出发异常 俩中功能时现代异常机制常用 3](#_Toc46)

[5.3. 异常传递 以检查异常 3](#_Toc2905)

[5.4. 错误有限设计思想 法神错误应该like报告 4](#_Toc19504)

[6. 变量 4](#_Toc340)

[7. 动态作用域 静态作用域 4](#_Toc24289)

[7.1. 作用域的演变 4](#_Toc25475)

[8. 类型 动态 静态类型 4](#_Toc301)

[8.1. 动态类型中变量结构体 obj 4](#_Toc23338)

[8.2. 类型推短 4](#_Toc21299)

[9. 代码归纳方法 5](#_Toc15434)

[9.1. Class 5](#_Toc9183)

[9.2. Pkg 5](#_Toc329)

[9.3. 模块module 5](#_Toc30333)

[9.4. 闭包 5](#_Toc20665)

[9.5. 散列表 5](#_Toc11759)

[10. 三大流程 循环 选择 顺序 5](#_Toc17985)

[11. 第 10 章　并发处理 5](#_Toc21085)

[12. 第 11 章　对象与类 6](#_Toc16385)

[13. 第 12 章　继承与代码再利用 7](#_Toc19867)

[13.2. 12.2　多重继承 的优点 适当使用 7](#_Toc15336)

# 程序设计是个什么概念呢？历史发展

## 连接电缆

大约半世纪以前，程序设计是个什么概念呢？

当时的程序设计就是指把这台计算机不同的端口通过电缆连接起来（图 2.1）。每次更改程序时

## 程序内置纸带打点

1949 年，EDSAC（爱达赛克，Electronic Delay Storage Automatic Calculator，电子延迟存储自动计算机）问世。这是一种通过纸带打点的方式来记录和读取数据的计算机（图 2.2）。程序作为数据通过纸带输入。不需要

## FORTRAN 语言问世

直到 1954 年，与大家现在使用的语言类似的程序设计语言才被发明出来。这就是 FORTRAN。 它的全称是 Formula Translating System（公式翻译系统）。现在，我们常用 X\* Y+Z

# 懒惰：程序员的三大美德之一

大家听说过“程序员的三大美德”吗？ Perl 语言的设计者 Larry Wall 在其著作 Programming Perl 中提出，优秀的程序员具有三大美德： 懒惰、急躁和傲慢（Laziness, Impatience and Hubris）。这就是俗称的程序员的三大美德。本节，我们介绍其中最重要的一项素质：懒惰。

懒惰（Laziness）

# [第 3 章　语法的诞生](https://read.douban.com/reader/ebook/12186524/toc/13" \t "https://read.douban.com/ebook/12186524/_blank)

##### [3.1　什么是语法](https://read.douban.com/reader/ebook/12186524/toc/14" \t "https://read.douban.com/ebook/12186524/_blank)

## [3.2　栈机器和 FORTH 语言](https://read.douban.com/reader/ebook/12186524/toc/15" \t "https://read.douban.com/ebook/12186524/_blank)

## [3.3　语法树和 LISP 语言](https://read.douban.com/reader/ebook/12186524/toc/16" \t "https://read.douban.com/ebook/12186524/_blank)

## [3.4　中缀表达式](https://read.douban.com/reader/ebook/12186524/toc/17" \t "https://read.douban.com/ebook/12186524/_blank) 前缀 后置表达式

## S表达式

# [第 5 章　函数](https://read.douban.com/reader/ebook/12186524/toc/27" \t "https://read.douban.com/ebook/12186524/_blank)

# 错误处理

## 返回值 vs 异常

异常rang vm来时刻检查返回值。。

## 可以追加错误类型和自著出发异常 俩中功能时现代异常机制常用

## 异常传递 以检查异常

检查异常的缺点太麻烦所以不普及

## 错误有限设计思想 法神错误应该like报告

# 变量

# 动态作用域 静态作用域

## [作用域的演变](https://read.douban.com/reader/ebook/12186524/toc/43" \t "https://read.douban.com/ebook/12186524/_blank)

# 类型 动态 静态类型

## 动态类型中变量结构体 obj

使用次数，类型，值

字符串

还有 大小，散列值，状态，值（第一。。。。滴n字符）

## 类型推短

# 代码归纳方法

## Class

## Pkg

## 模块module

## 闭包

## 散列表

# 三大流程 循环 选择 顺序

# [第 10 章　并发处理](https://read.douban.com/reader/ebook/12186524/toc/62" \t "https://read.douban.com/ebook/12186524/_blank)

##### [10.1　什么是并发处理](https://read.douban.com/reader/ebook/12186524/toc/63" \t "https://read.douban.com/ebook/12186524/_blank)

##### [10.2　细分后再执行](https://read.douban.com/reader/ebook/12186524/toc/64" \t "https://read.douban.com/ebook/12186524/_blank)

##### [10.3　交替的两种方法](https://read.douban.com/reader/ebook/12186524/toc/65" \t "https://read.douban.com/ebook/12186524/_blank)

##### [10.4　如何避免竞态条件](https://read.douban.com/reader/ebook/12186524/toc/66" \t "https://read.douban.com/ebook/12186524/_blank)

##### [10.5　锁的问题及对策](https://read.douban.com/reader/ebook/12186524/toc/67" \t "https://read.douban.com/ebook/12186524/_blank)

# [第 11 章　对象与类](https://read.douban.com/reader/ebook/12186524/toc/69" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.1　什么是面向对象](https://read.douban.com/reader/ebook/12186524/toc/70" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.2　归集变量与函数建立模型的方法](https://read.douban.com/reader/ebook/12186524/toc/71" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.3　方法 1：模块、包](https://read.douban.com/reader/ebook/12186524/toc/72" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.4　方法 2：把函数也放入散列中](https://read.douban.com/reader/ebook/12186524/toc/73" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.5　方法 3：闭包](https://read.douban.com/reader/ebook/12186524/toc/74" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.6　方法 4：类](https://read.douban.com/reader/ebook/12186524/toc/75" \t "https://read.douban.com/ebook/12186524/_blank)

##### [11.7　小结](https://read.douban.com/reader/ebook/12186524/toc/76" \t "https://read.douban.com/ebook/12186524/_blank)

# [第 12 章　继承与代码再利用](https://read.douban.com/reader/ebook/12186524/toc/77" \t "https://read.douban.com/ebook/12186524/_blank)

##### [12.1　什么是继承](https://read.douban.com/reader/ebook/12186524/toc/78" \t "https://read.douban.com/ebook/12186524/_blank)

## [12.2　多重继承](https://read.douban.com/reader/ebook/12186524/toc/79" \t "https://read.douban.com/ebook/12186524/_blank) 的优点 适当使用

##### [12.3　多重继承的问题——还是有冲突](https://read.douban.com/reader/ebook/12186524/toc/80" \t "https://read.douban.com/ebook/12186524/_blank)

##### [12.4　小结](https://read.douban.com/reader/ebook/12186524/toc/81" \t "https://read.douban.com/ebook/12186524/_blank)

代码之髓：编程语言核心概念 - 〔日〕西尾泰和 \_ 豆瓣阅读.mhtml