程序设计语言原理

THE PRINCIPLES OF PROGRAMMING LANGUAGE

先修课程：C语言、编译原理、离散数学

主讲：吕卫锋
LWF@NLSDE.BUAA.EDU.CN;

诸彤宇
ZHUTONGYU@BUAA.EDU.CN;

程序设计语言原理

- **课程名称：程序语言设计方法学**

  The Principles Of Programming Language

  学时学分：48学时

  先修课程：C语言、编译原理、离散数学

- **主讲：** 吕卫锋 lwf@nlsde.buaa.edu.cn;

  诸彤宇 zhutongyu@nlsde.buaa.edu.cn;

- **课件网址：** http://pl.nlsde.buaa.edu.cn/

  http://course.buaa.edu.cn

考核

- 考试：50%
- 作业：
  - 平时作业：10%
  - 大作业：设计一个计算机语言，40%

大作业的要求

- 组队：每组不多于3人
- 提交设计报告

1. 新定义语言的背景和目标

   设计驱动，基础范型，参考语言及其不同点，围绕设计准则方面的考虑

2. 语法设计

   语言要素，静态/动态、编译/解释、跨平台等方面的考虑；数据类型、关键字、Token对象等词法规则考虑；BNF（举例用图来表达）、抽象语法树、语法分析等的设计考虑

3. 涉及范型的设计

   控制流相关的设计（分支、迭代等）；对象、并发机制、闭包等

4. 典型语言机制的语义相关的证明（举例说明）

5. 与对标语言在实现上的差异说明

   语言差异，运行差异

6. 验证与测试

每组评优的条件

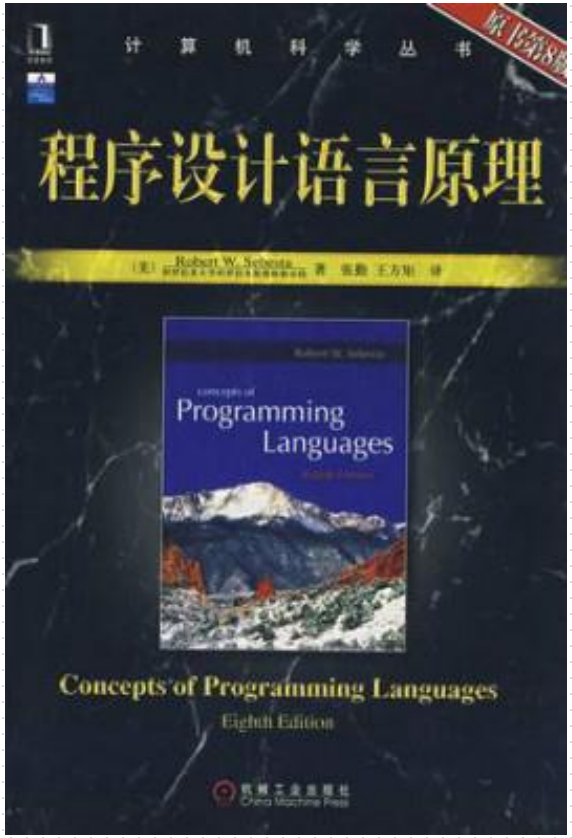
- 每组不多于2人
- 最多给12组
- 考试时完成B卷
- 考核标准：
  - 在普通作业标准基础上：
  - 课堂讲解（PPT）
  - 可运行、有测试样例

学好这门课的难点

- 贯穿计算软件体系的课程
  - 理论很多
  - 知识点很多
- 先修内容要求较高，理论与工程实践相结合
- 理解设计
- 主线与要点相结合

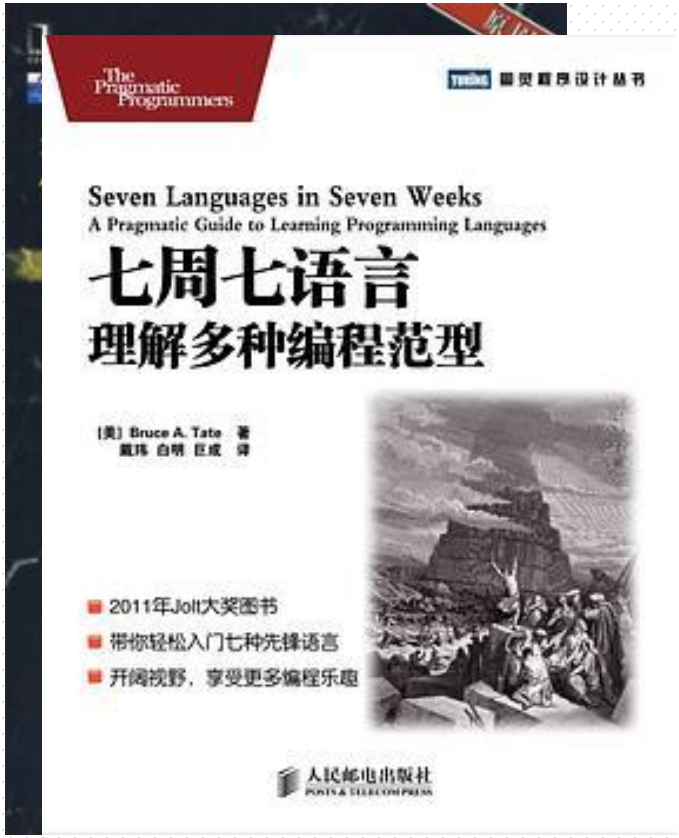
参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)



参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)

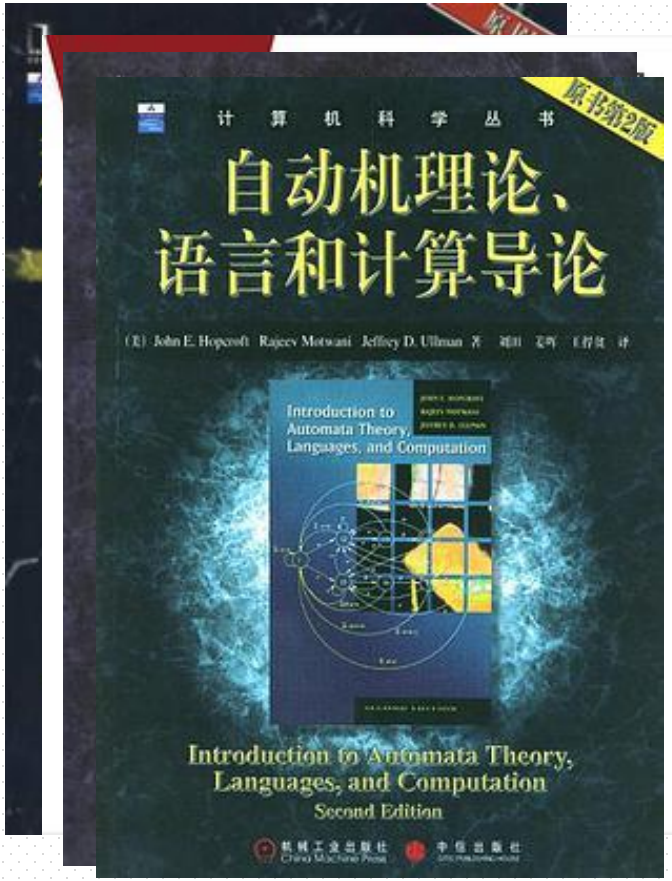


参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义

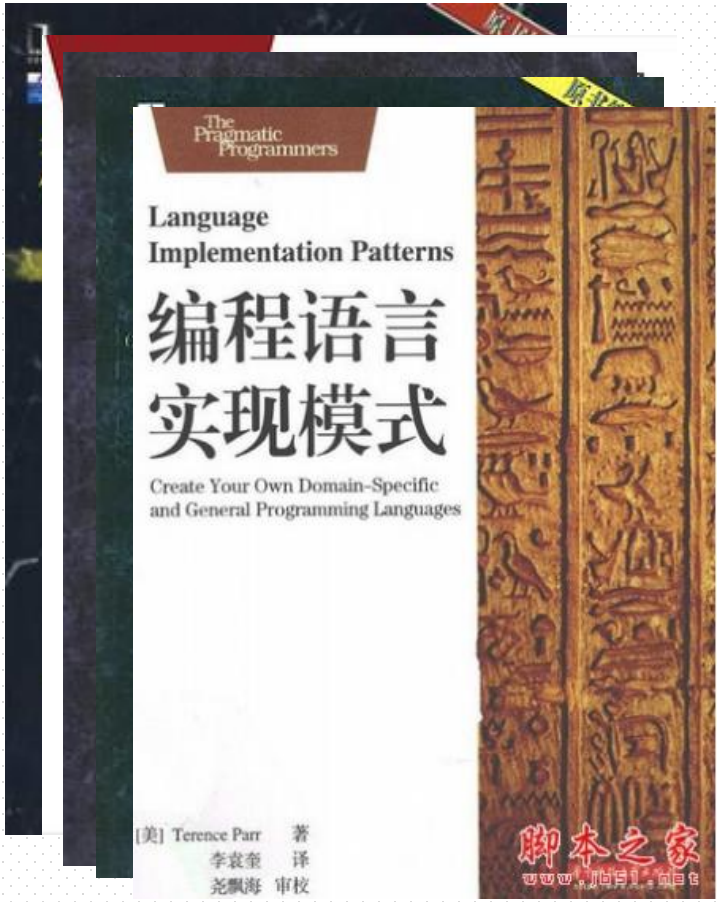
参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义
- 自动机理论、语言和计算导论



参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义
- 自动机理论、语言和计算导论
- 编程语言实现模式

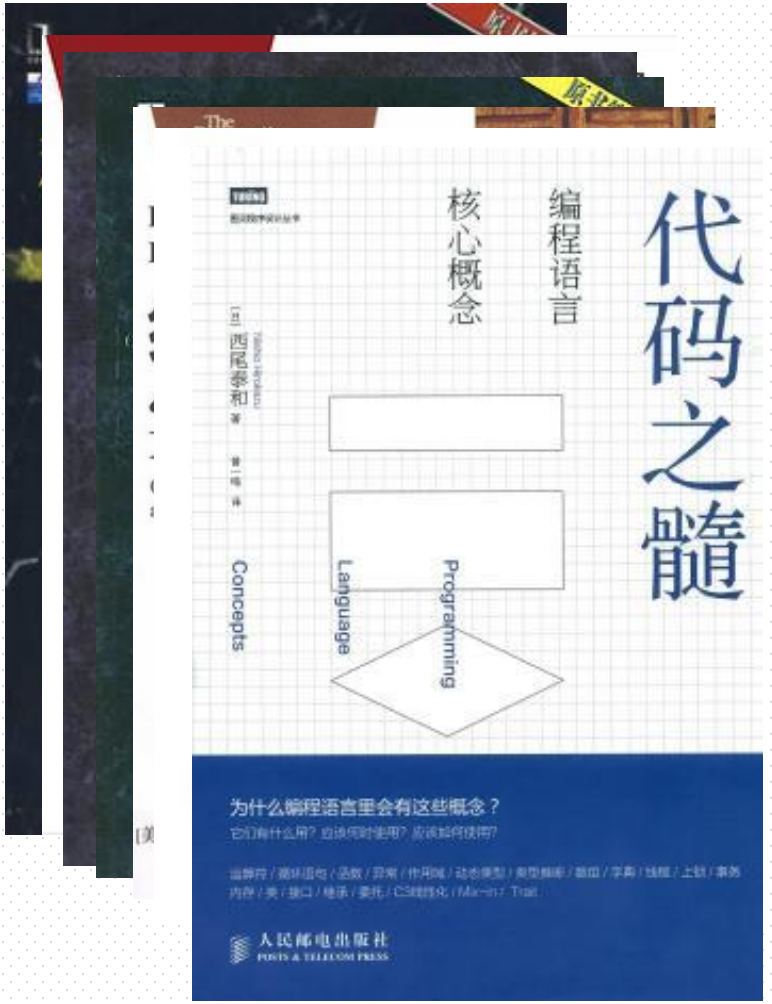


参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义
- 自动机理论、语言和计算导论
- 编程语言实现模式
- 代码的未来

参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义
- 自动机理论、语言和计算导论
- 编程语言实现模式
- 代码的未来
- 代码之髓



参考书目

- 程序设计语言原理(北航)
- 程序设计语言原理(第八版)
- 七周七语言(理解多种编程范型)
- 程序设计语言的形式语义
- 自动机理论、语言和计算导论
- 编程语言实现模式
- 代码的未来
- 代码之髓
- 两周自制脚本语言

第一章 计算机语言的学科形态与发展历程

- 计算机语言在计算学科中占有特殊的地位，它是计算学科中最富有智慧的成果之一，它深刻地影响着计算学科各个领域的发展。不仅如此，计算机语言还是程序员与计算机交流的主要工具。因此，可以说如果不了解计算机语言，就谈不上对计算学科的真正了解。

计算学科中的抽象形态

- 《计算作为一门学科》报告认为：理论、抽象和设计是我们从事本领域工作的3种主要形态。

- 按人们对客观事物认识的先后次序，抽象列为第一个学科形态，理论列为第二个学科形态。抽象源于实验科学。按客观现象的研究过程，抽象形态包括以下4个步骤的内容：

  （1）形成假设；

  （2）建造模型并作出预测；

  （3）设计实验并收集数据；

  （4）对结果进行分析。

计算学科中的理论形态

- 在计算学科中，从统一合理的理论发展过程来看，理论形态包括以下4个步骤的内容：

  （1）表述研究对象的特征（定义和公理）；

  （2）假设对象之间的基本性质和对象之间可能存在的关系（定理）；

  （3）确定这些关系是否为真（证明）；

  （4）结论。

计算学科中的设计形态

- 在计算学科中，从为解决某个问题而实现系统或装置的过程来看，设计形态包括以下4个步骤的内容：

（1）需求分析；

（2）建立规格说明；

（3）设计并实现该系统；

（4）对系统进行测试与分析。

计算学科中3个学科形态内在联系

3个学科形态的内在联系

- 抽象源于现实世界。建立对客观事物进行抽象描述的方法。建立具体问题的概念模型，实现对客观世界的感性认识。
- 理论源于数学。建立完整的理论体系。建立具体问题的数学模型，从而实现对客观世界的理性认识。
- 设计源于工程。对客观世界的感性认识和理性认识的基础上，完成一个具体的任务；对工程设计中所遇到的问题进行总结，提出问题，由理论界去解决它。

目录

一、形式语言、图灵机与冯·诺依曼型计算机

形式语言的基本特点

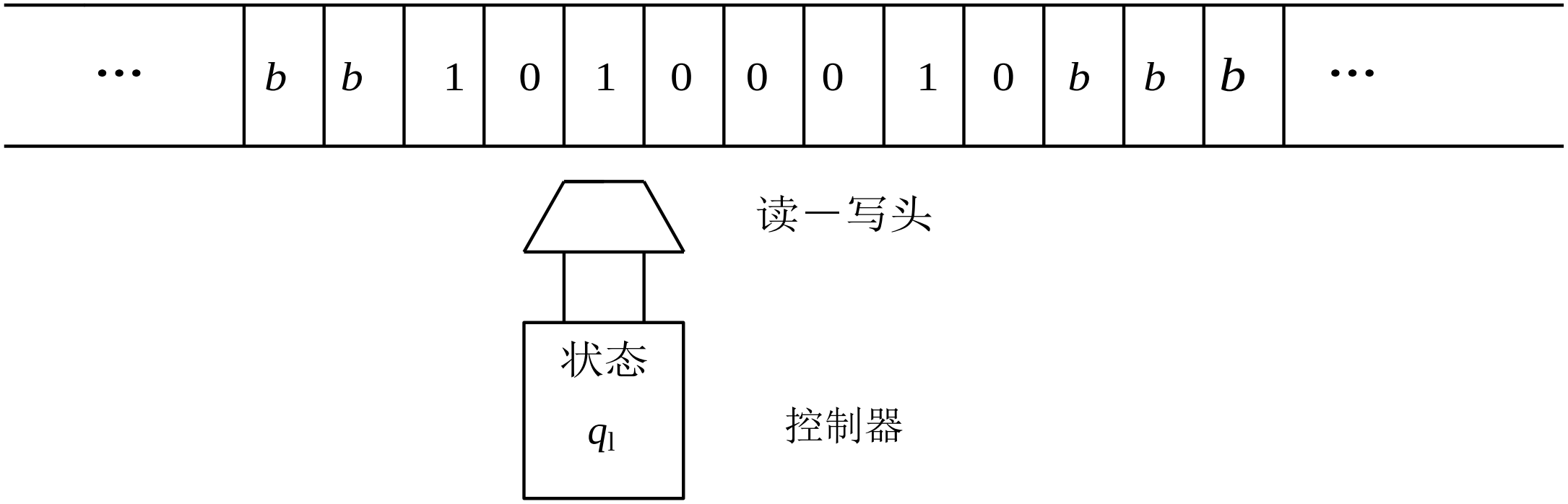
- 有一组初始的、专门的符号集；
- 有一组精确定义的，由初始的、专门的符号组成的符号串转换成另一个符号串的规则。
- 在形式语言中，不允许出现根据形成规则无法确定的符号串。

形式语言的语法

- 形式语言的语法：形式语言中的转换规则。
  - 语法不包含语义。
  - 在一个给定的形式语言中，可以根据需要，通过赋值或模型对其进行严格的语义解释，从而构成形式语言的语义。
  - 语法和语义要作严格的区分。

图灵机

- 图灵机由一条两端可无限延长的带子、一个读写头以及一组控制读写头工作的命令组成



一个给定机器的“程序”

- 机器内的五元组（$q_i S_j S_k R$（或L或N）q_l）形式的指令集，五元组定义了机器在一个特定状态下读入一个特定字符时所采取的动作。5个元素的含义如下：
  - q_i表示机器目前所处的状态；
  - S_j表示机器从方格中读入的符号；
  - S_k表示机器用来代替S_j写入方格中的符号；
  - R、L、N分别表示向右移一格、向左移一格、不移动；
  - q_l表示下一步机器的状态。

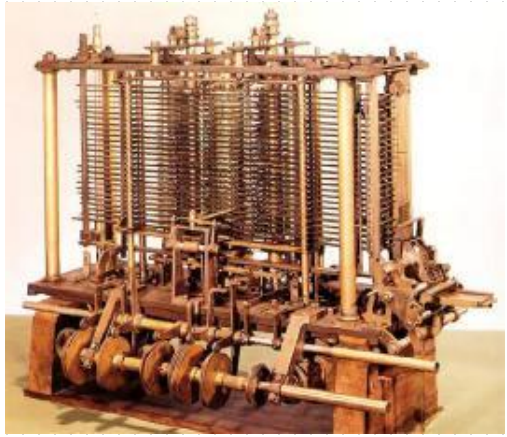
图灵机及其他计算模型

- 图灵的观点及结论：
  - 凡是能用算法方法解决的问题，也一定能用图灵机解决；凡是图灵机解决不了的问题，任何算法也解决不了。
- 与图灵机等价的计算模型：
  - 递归函数
  - λ-演算
  - POST规范系统
- 图灵机是从过程这一角度来刻画计算的本质，其结构简单、操作运算规则也较少，从而为更多的人所理解。

图灵机的计算能力

- 图灵机可以计算
  - $S(x) = x + 1$（后继函数），
  - $N(x) = 0$（零函数），
  - $U_i^{(n)}(x_1, x_2, \ldots, x_n) = x_i$，$1 \le i \le n$（投影函数）
  - 上述3个函数的任意组合。
- 从递归论中，我们知道这3个函数属于初始递归函数，
  - 任何原始递归函数都是从这3个初始递归函数经有限次的复合、递归和极小化操作得到的。
  - 从可计算理论可知每一个原始递归函数都是图灵机可计算的。

“电子”计算机之前

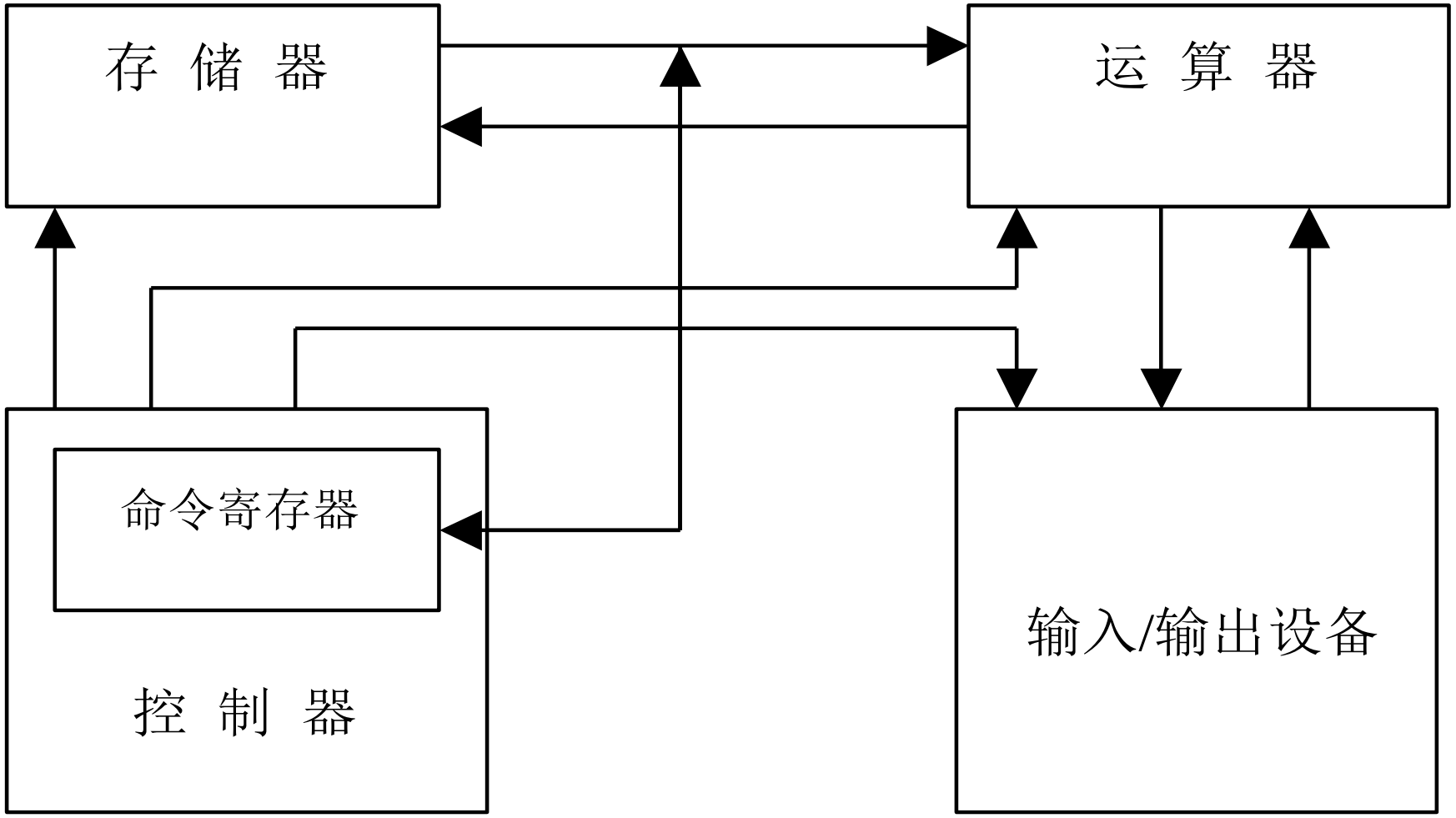


- Charles Babbage和Augusta Ada Byron
  - 第一台计算器、首个算法和程序
- Zuse计算机(**Konrad Zuse)**
  - Z - 1：第一次采用了二进制数，纯机械式，穿孔带
  - Z - 2：电磁式计算机
  - Z - 3：2600个继电器
  - Z - 4：1945年，存储器单元也从64位扩展到1024位。为了使机器的效率更高，Zuse设计了一种编程语言Plankalkuel

冯·诺依曼型计算机

- ENIAC的结构在很大程度上是依照机电系统设计的，还存在重大的线路结构等问题。
- 在图灵等人工作的影响下，1946年6月，美国杰出的数学家冯·诺依曼（Von Neumann）及其同事完成了关于“电子计算装置逻辑结构设计”的研究报告，
  - 具体介绍了制造电子计算机和程序设计的新思想
  - 至今为止，大多数计算机采用的仍然是冯·诺依曼型计算机的组织结构，只是作了一些改进而已。因此，冯·诺依曼被人们誉为“计算机器之父”。
- UNIVAC I

冯·诺依曼型计算机的组织结构



指令系统

- CPU必须能够解码并执行的机器指令很少
  - 一旦计算机可以执行一些基本的而且是精选的操作，加入额外的操作，理论上是不会改变计算机能力
- 是否充分利用这种特性导致了两种不同的计算机设计：
  - CISC(reduced instruction set computer)
  - RISC(complex instruction set computer)

机器指令

- 机器指令系统——每台数字电子计算机在设计中，都规定了一组指令。
- 机器语言——用机器指令形式编写的程序。
- 在裸机级，计算机语言关于算法的描述采用的是实际机器的机器指令，它的符号集是{0，1}，
  - 支撑实际机器的理论是图灵机等计算模型；
  - 在图灵机等计算模型理论的指导下，有关设计形态的主要成果有冯·诺依曼型计算机等具体实现思想和技术，以及各类数字电子计算机产品。

计算机语言在裸机级所取得的主要成果

计算机语言	抽象	理论	设计
裸机级的主要内容和成果	语言的符号集为：{0，1}；用机器指令对算法进行描述	图灵机(过程语言的基础)、波斯特系统(字符串处理语言的基础)、λ-演算(函数式语言的基础)等计算模型	冯·诺依曼型计算机等实现技术；数字电子计算机产品

二、高级语言出现、发展与主要学科形态

汇编语言

- 采用字符和十进制数来代替二进制代码的思想。
- 例3.10 对2+6进行计算的算法描述
  - 用机器指令对“2+6”进行计算的算法描述：

    1011000000000110

    0000010000000010

    10100010010100000000000000
  - 汇编语言对“2+6”进行计算的算法描述：

    MOV AL，6

    ADD AL，2

    MOV VC，AL

汇编语言

- 汇编语言语句与特定的机器指令有一一对应的关系，但是它毕竟不同于由二进制组成的机器指令，它还需要经汇编程序翻译为机器指令后才能运行。
- 汇编语言源程序经汇编程序翻译成机器指令，再在实际的机器中执行。
  - 就汇编语言的用户而言，该机器是可以直接识别汇编语言的，从而产生了一个属于抽象形态的重要概念，即虚拟机的概念。

虚拟机

- 抽象的计算机
  - 由软件实现，并与实际机器一样，都具有一个指令集并可以使用不同的存储区域。
  - 例如，一台机器上配有C语言和Pascal语言的编译程序，对C语言用户来说，这台机器就是以C语言为机器语言的虚拟机，对Pascal用户来说，这台机器就是以Pascal语言为机器语言的虚拟机。

虚拟机的层次之分

- 虚拟机可分为
  - 固件虚拟机
  - 操作系统虚拟机
  - 汇编语言虚拟机
  - 高级语言虚拟机
  - 应用语言虚拟机等

虚拟机的意义和作用

- 当机器（实际机器或虚拟机）确定下来后，所识别的语言也随之确定；反之，当一种语言形式化后，所需要支撑的机器也可以确定下来。从计算机系统的层次结构图中可以清晰地看到这种机器与语言的关系。虚拟机是计算学科中抽象的重要内容。引入虚拟机的概念，就计算机语言而言，有以下意义和作用：
- 有助于我们正确理解各种语言的实质和实现途径
- 推动了计算机体系结构以及计算机语言的发展
- 有助于各层次计算机语言自身的完善

高级语言

- 虽然与机器语言相比，汇编语言的产生是一个很大的进步，但是用它来进行程序设计仍然比较困难。于是人们着手对它进行改进。一是发展宏汇编，即用一条宏指令代替若干条汇编指令，从而提高编程效率。现在人们使用的汇编语言，大多数都是宏汇编语言。二是创建高级语言，使编程更加方便。
  - 如用高级语言对例子2+6进行计算的算法描述，其描述与数学描述一样，即2+6。

高级语言的分类

按语言的特点，可以将高级语言划分为：

- 过程式语言（如Cobol，Forturn，Algol，Pascal，Ada，C）
- 函数式语言（如Lisp）
- 数据流语言（如SISAL，VAL）
- 面向对象语言（如Smalltalk，CLU，C++）
- 逻辑语言（如Prolog）
- 字符串语言（如SNOBOL）
- 并发程序设计语言（如Concurrent Pascal，Modula 2，Go）等

高级语言的形式化

20世纪50年代

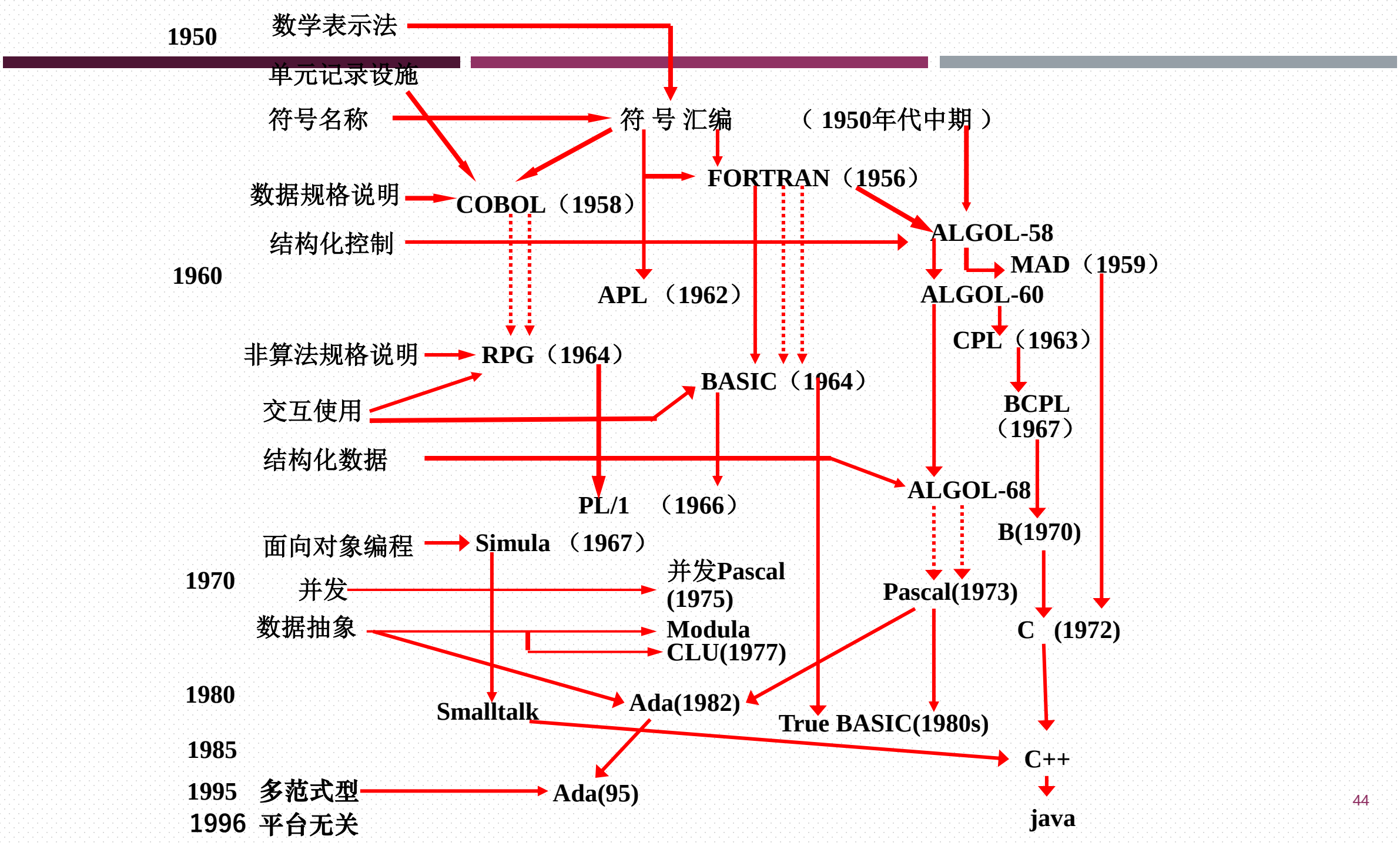
- 美国语言学家乔姆斯基（Noam Chomsky）关于语言分层的理论，
- 巴科斯（Backus）、瑙尔（Naur）的关于“上下文无关方法表示形式”的研究成果推动了语法形式化的研究。
- 其结果是，在ALGOL60的文本设计中第一次使用了BNF范式来表示语法，并且第一次在语言文本中明确提出应将语法和语义区分开来。

高级语言的形式化

- 20世纪50年代至60年代间，面向语法的编译自动化理论得到了很大发展，使语法形式化研究的成果达到实用化的水平。
- 语法形式化问题基本解决以后，人们逐步把注意力集中到语义形式化的研究方面, 20世纪60年代，相继诞生了
  - 操作语义学
  - 指称语义学
  - 公理语义学
  - 代数语义学等语义学理论

高级语言简史

- 50年代高级语言出现
- 60年代奠基性研究
- 70年代完善的软件工程工具
- 80年代面向对象发展
- 90年代多范型、持久化、多媒体、平台无关
- 并行、声明式程序设计时代


1950
数学表示法
单元记录设施
符号名称
符 号 汇编
（ 1950年代中期 ）
数据规格说明
COBOL（1958）
FORTRAN（1956）
结构化控制
ALGOL-58
MAD（1959）
1960
APL （1962）
ALGOL-60
CPL（1963）
非算法规格说明
RPG（1964）
BASIC（1964）
交互使用
BCPL
（1967）
结构化数据
ALGOL-68
PL/1 （1966）
B(1970)
面向对象编程
Simula （1967）
1970
并发
并发Pascal
(1975)
Pascal(1973)
数据抽象
Modula
CLU(1977)
C (1972)
1980
Smalltalk
Ada(1982)
True BASIC(1980s)
1985
C++
1995
多范式型
Ada(95)
1996
平台无关
java

二十一世纪语言趋势

- 并行：如何做并行程序设计是每个计算工作者的挑战
  - 狭义摩尔定律失效
  - 并行系统的可靠性
  - 并行系统开发效率
- 脚本语言：计算机应用的崛起
  - WEB服务端：PHP，ASP，JSP等
  - Web客户端：Javascipt
  - 广泛应用开发：Perl，Python，Ruby
  - 专用脚本语言：Tcl/tk
  - 丰富、灵活、快速开发、解释执行、社区
- 软件设计：基于组件、服务、AOP的软件开发

PLATFORM-BASED DEVELOPMENT (PBD)

随着面向特定平台的编程越来越广泛，例如：WEB和移动设备；

基于特定平台的开发，需要考虑特定的硬件、API和特殊的上下文。其与通用编程有所不同。

桌面操作系统	移动操作系统	软件框架	硬件
AmigaOS, AmigaOS 4	Android	Binary Runtime Environment for Wireless (BREW)	Commodity computing platforms • Wintel • Macintosh • ARM architecture • x86 with Unix-like systems
FreeBSD, NetBSD, OpenBSD	Bada	Cocoa	
IBM i	BlackBerry OS	Cocoa Touch	
Linux	Firefox OS	Common Language Infrastructure (CLI): Mono, .NET Framework, Silverlight	
Microsoft Windows	iOS	Flash: AIR	
OpenVMS	Embedded Linux	GUN	Video game consoles, any variety • PlayStation • Xbox, • Nintendo
Classic Mac OS	Palm OS	Java platform	
macOS	Symbian	LiveCode	
OS/2	Tizen	Microsoft XNA	
Solaris	WebOS	Mozilla Prism, XUL and XULRunner	RISC processor based machines running Unix variants
Tru64 UNIX	LuneOS	Open Web Platform	
VM	Windows Mobile	Oracle Database	
QNX	Windows Phone	Qt	
z/OS		Universal Windows Platform ： Windows Runtime	

高级语言中抽象、理论和设计形态

抽象	理论	设计
常用的符号：数字(0~9)，大小写字母(A~Z、a~z)，括号，运算符(+，−，*，/)等; 用高级语言对算法进行的描述; 语言的分类方法; 各种数据类型的抽象实现模型; 词法分析、编译、解释和代码优化的方法; 词法分析器、扫描器、编译器组件和编译器的自动生成方法	形式语言和自动机理论; 形式语义学：操作、指称、公理、代数、并发和分布式程序的形式语义	特定语言：过程式的COBOL，FORTURN，ALGOL，Pascal，Ada，C），函数式的（LISP），数据流的（SISAL，VAL），面向对象的（Smalltalk，C++），逻辑的（Prolog），字符串（SNOBOL），和并发（Concurrent Pascal，Modula 2）等语言; 词法分析器和扫描器的产生器（如YACC，LEX），编译器产生器; 语法和语义检查，成型、调试和追踪程序

三、面向应用的计算语言发展与学科形态

计算机语言的划分

一般将它划分为5代：

- 第一代为机器语言；
- 第二代为汇编语言；
- 第三代为高级语言；
- 第四代为“非过程性语言”；
- 第五代为自然语言。

4GL

- 提供了功能强大的非过程化问题定义手段，用户只需告知系统**“做什么”**，而无需说明**“怎么做”**
- **功能核心包括：数据库管理、报告生成、数值优化、图形界面开发、Web开发**，进一步构造了开发高层软件系统的开发环境。
  - 数据处理：SQL、Adabas、Rust等；
  - 报告生成：Progress 4GL Query/Result等；
  - 数值优化：MATLAB等；
  - 移动应用：Go、Python、Swift等；

应用语言中有关抽象、理论和设计形态的主要内容

抽象	理论	设计
采用应用语言对算法进行描述	特定应用领域的支撑理论：数据库、数值优化、机器学习、大数据计算等领域的支撑理论	面向特定平台或领域应用的语言和编程框架：Go、Python、R、Matlab、TensorFlow、Spark、Flink、Sprin、Kubernetes、LCNC。

应用语言中有关抽象、理论和设计形态的主要内容

- 对计算机语言抽象、理论和设计3个学科形态的研究，有助于我们正确理解计算机语言的本质，以及更好地把握它的研究方向，从而能更好地进行计算学科的研究。

四、程序设计语言分类与典型语言

语言分类

- 机器依赖
  - 低、高、中
- 应用领域

商用	科学	系统	模拟	实时
COBOL 各种表单语言 C, C++	FORTRAN	BLISS C, C++	GPSS SIMULA	CHILL GYPSY
嵌入式	人工智能	处理命令	教学	正文
Ada	Prolog LISP	Sell TCL PERL	Pascal BASIC LOGO	SNOBOL Word
打印出版	串、数组、表	数据库		
Postscript TeX	APL SNOBOL LISP	SQL		

续

- 程序范型
  - 单范型/多范型
    - 命令式（过程式） FORTRAN、COBOL、PL/1、PASCAL、Ada-83
    - 面向对象 Smalltalk、Java、Eiffel
    - 数据流 Val
    - 函数式 ML LISP
    - 逻辑式 Prolog
    - 关系式 SQL
  - 多范型Nail (1983) G (1986) C++、Ada-95 、CLOS

重要语言20种

- FORTRAN、COBOL、ALGOL、PL/1
- LISP、ALGOL-68、BASIC、PASCAL
- APL、Ada、Simula、C/C++
- Smalltalk、ML、Prolog、SQL、Scala
- Java、XML、Python、Go、Rust

课程编程语言要求

熟悉Go、Python、Scala三种语言

- 掌握形式化的描述语言的语法特征
- 了解三种语言的主要特征和机制
- 熟练使用三种语言实现程序设计与软件开发

GO语言学习社区

- 官方网站：https://golang.org/
- 基础学习：

  http://c.biancheng.net/golang/

  https://www.runoob.com/go/go-tutorial.html
- 进阶学习：

  https://chai2010.cn/advanced-go-programming-book/
- 参考书籍：

  《Go 入门指南》https://github.com/unknwon/the-way-to-go_ZH_CN

PYTHON语言学习社区

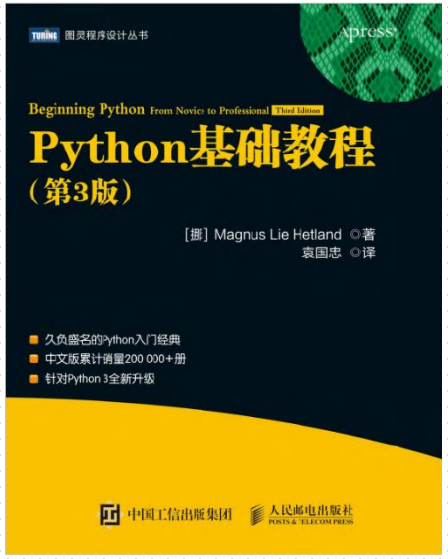
- **Python**

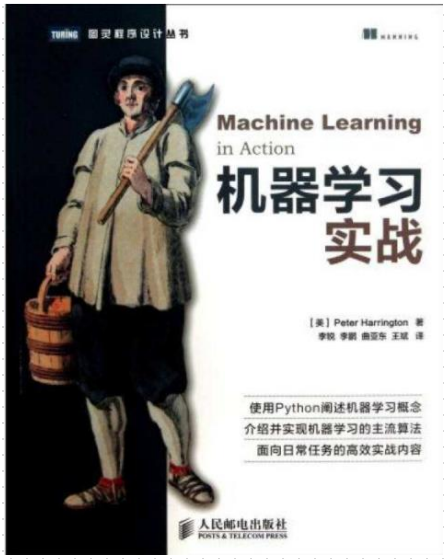
官网：https://www.python.org/

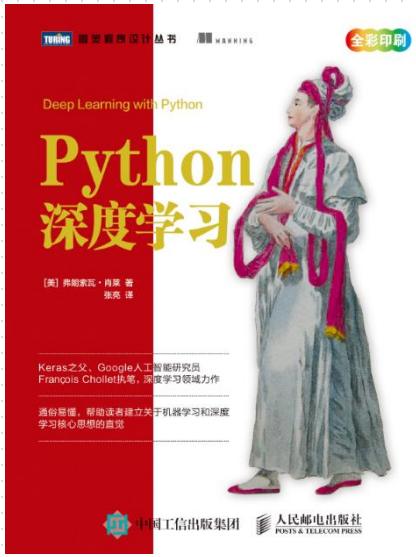
基础学习： http://www.runoob.com/python/python-tutorial.html

机器学习与深度学习： http://scikit-learn.org/stable/

https://morvanzhou.github.io/







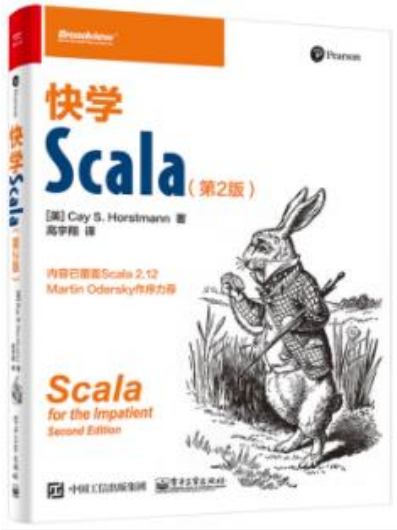
SCALA学习社区

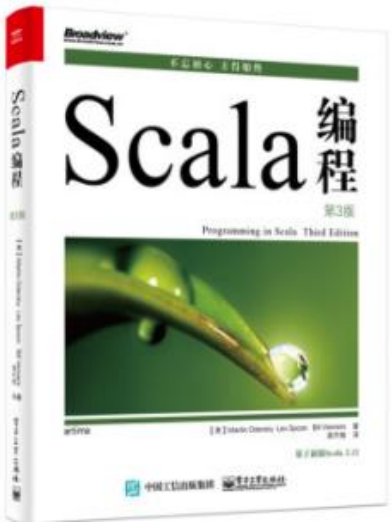
- **Scala**

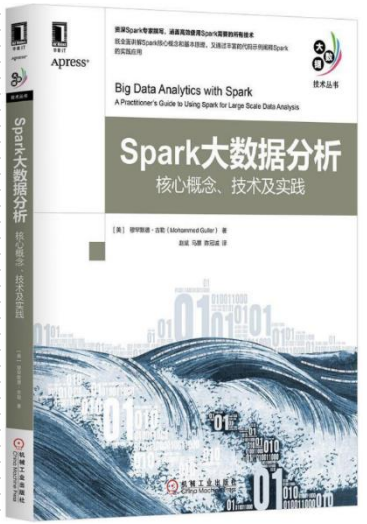
  官网：https://www.scala-lang.org/

  基础学习：http://www.runoob.com/scala/scala-tutorial.html

  大数据编程：https://www.ibm.com/developerworks/cn/opensource/os-cn-spark-practice1/index.html?lnk=hm







作业

- 画出21世纪的新语言图
- 此作业在大作业中体现，根据新世纪语言的特点，推导出你的语言的设计驱动（word1~2页左右）。