编译原理概述

魏恒峰

hfwei@nju.edu.cn

2022 年 10 月 31 日 (周一)

7 周 $=$ 14 次课 $<$ 8 周 $=$ 16 次课

作业 (0 分): ≈ 7 次作业, 每周 $1 \sim 2$ 题

作业 (0 分): ≈ 7 次作业, 每周 $1 \sim 2$ 题

实验 (60 分): ≈ 8 次必做实验 $+ 1$ 次选做实验 (≤ 5 分)

作业 (0 分): ≈ 7 次作业, 每周 $1 \sim 2$ 题

实验 (60 分): ≈ 8 次必做实验 $+ 1$ 次选做实验 (≤ 5 分)

期末测试 (40 分): 考试周统一安排; 2 小时; 开卷

作业 (0 分): ≈ 7 次作业, 每周 $1 \sim 2$ 题

实验 (60 分): ≈ 8 次必做实验 $+ 1$ 次选做实验 (≤ 5 分)

期末测试 (40 分): 考试周统一安排; 2 小时; 开卷

附加作业 (≤ 5 分): 报告 + 录屏方式, 学习更现代的编译原理与技术

附加项: 不计分

期末测试: 交与教务处

实验: 当次实验计零分, 并扣 5 分总评

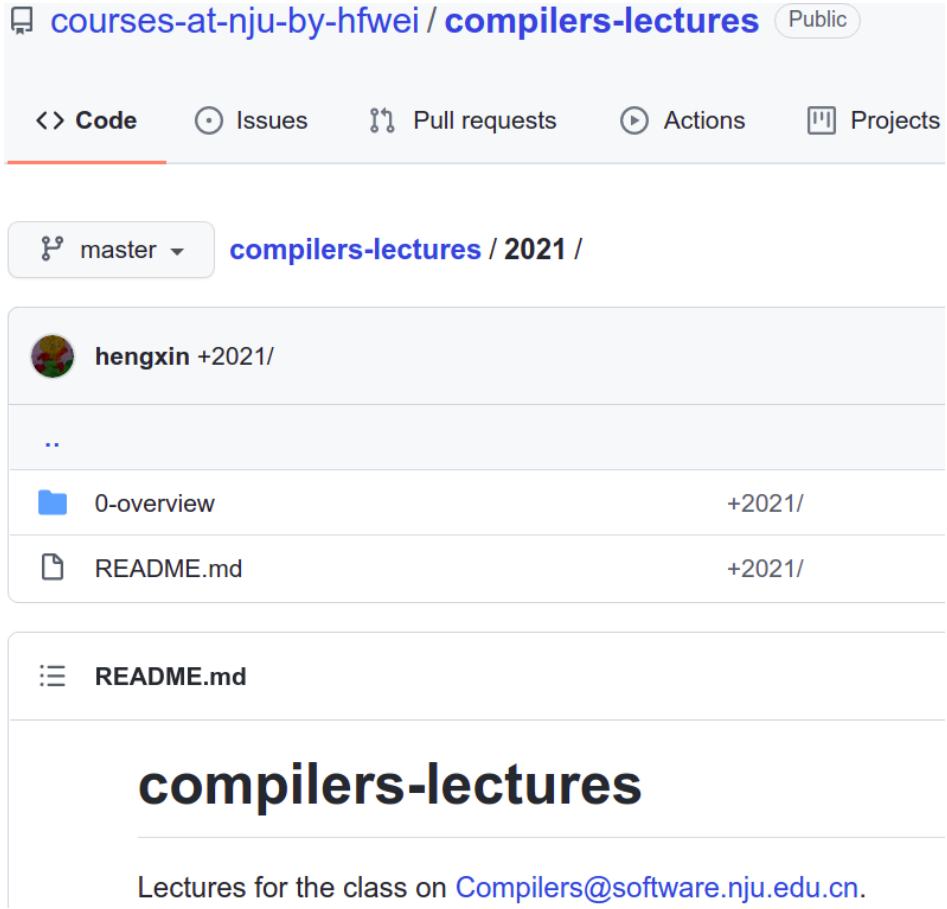
邀请码：JCZ837HE

每周三晚上布置作业　　下周三 23 : 55 前提交作业

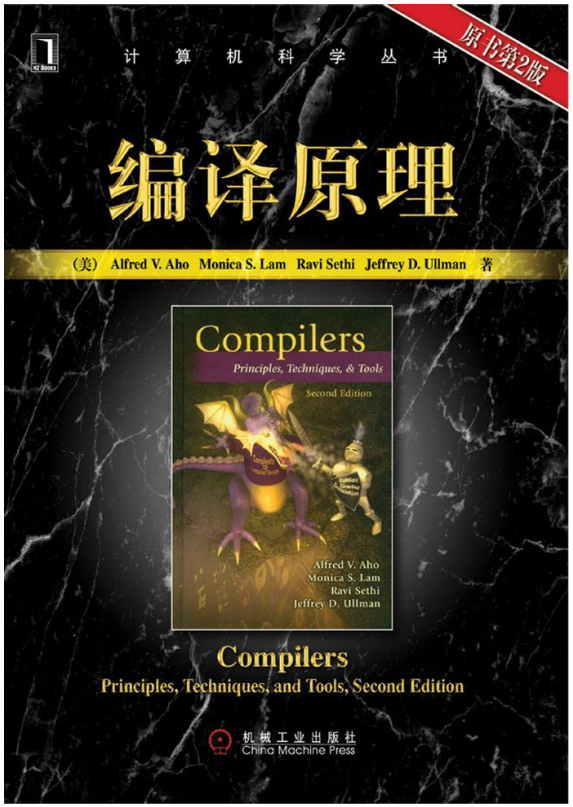
QQ 群号：**711805817**

助教：夏宇、潘煜光、顾龙、...

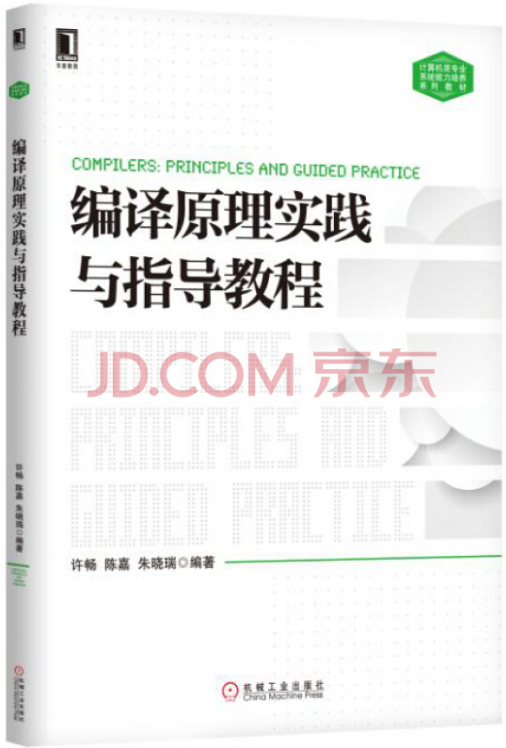
QQ 验证：**2021-编译原理**



https://github.com/courses-at-nju-by-hfwei/compilers-lectures/tree/master/2021



也可使用“**本科教学版**”



https://cs.nju.edu.cn/changxu/2_compiler/index.html

Flex: 词法分析器生成器

Bison: 语法分析器生成器

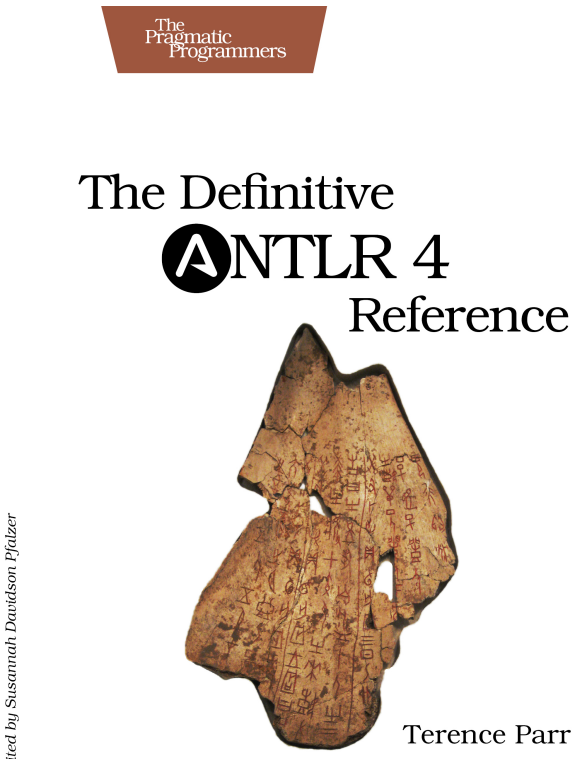
Flex: 词法分析器生成器

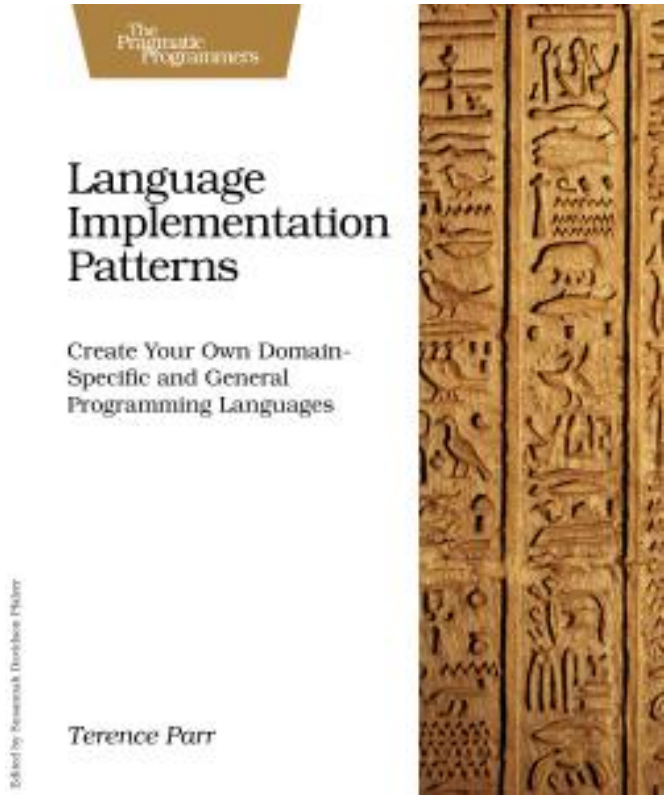
Bison: 语法分析器生成器

不够现代, 本课程不再支持

Terence Parr

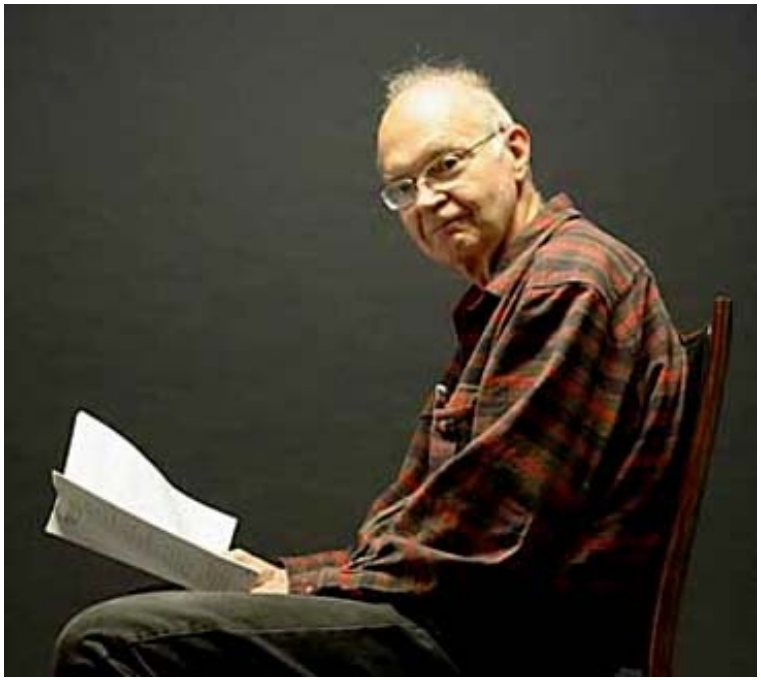
https://www.antlr.org/index.html

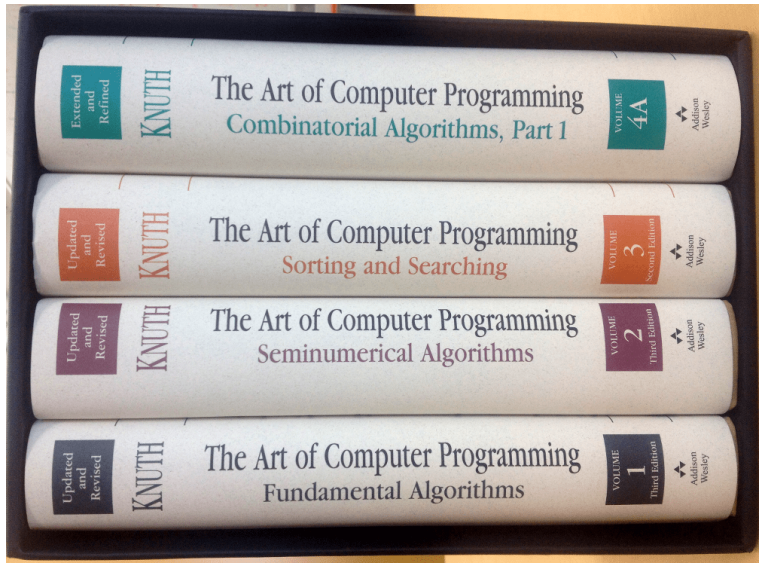

The Pragmatic Programmers
The Definitive
ANTLR 4
Reference
Terence Parr
Edited by Susannah Davidson Pfalzer


The Pragmatic Programmers
Language Implementation Patterns
Create Your Own Domain-Specific and General Programming Languages
Terence Parr


welcome
again

the “father of the analysis of algorithms”

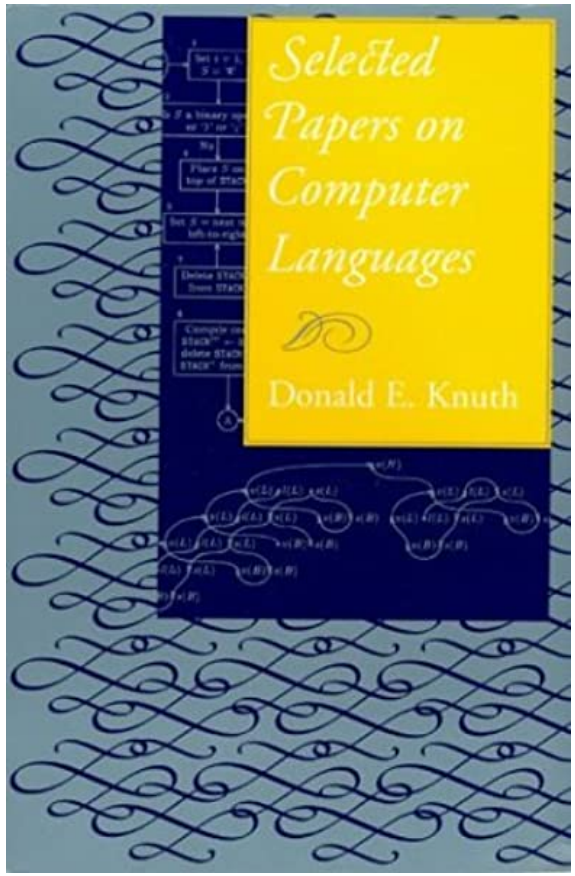




Donald E. Knuth (1938 $\sim$)

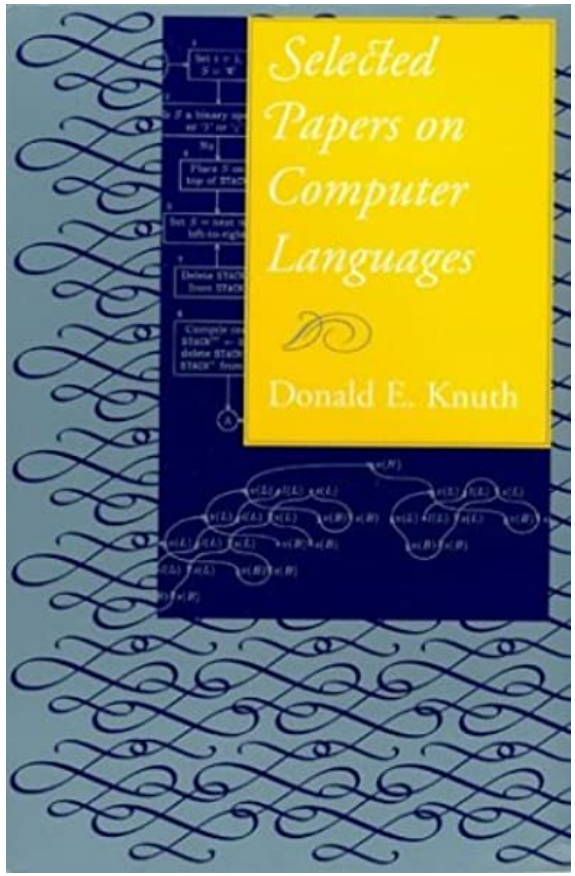
Turing Award, 1974

"Selected Papers on Computer Languages"



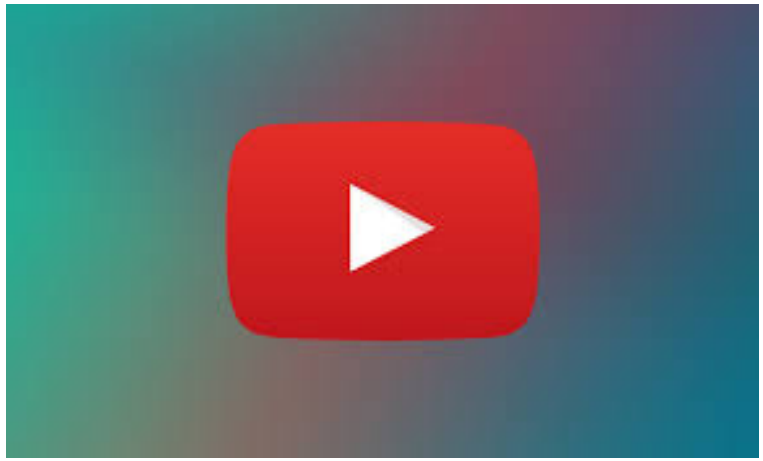
LR Parser (语法) **Attribute Grammar (语义)**

"Selected Papers on Computer Languages"



LR Parser (语法) **Attribute Grammar (语义)**

ALGOL 58 Compiler



"I got a job at the end of my senior year
to write a compiler for Burroughs"

“高级”语言 $\Longrightarrow$ (通常)“低级”语言 (如, 汇编语言)

汇编语言经过**汇编器**生成机器语言

“高级”语言 $\Longrightarrow$ (通常)“低级”语言 (如, 汇编语言)

汇编语言经过**汇编器**生成机器语言

GopherJS - A compiler from Go to JavaScript

godoc reference | used by 1.2k projects | PASSED

GopherJS compiles Go code (golang.org) to pure JavaScript code. Its main purpose is to give you the opportunity to write front-end code in Go which will still run in all browsers.

Q：机器语言是如何跑起来的?

Q：机器语言是如何跑起来的?

作业 (P1 ∼ P8): https://www.bilibili.com/video/BV1EW411u7th

语言类应用程序

- 配置文件解析 (.properties)
- CSV 文件 (Comma-Separated Values)
- JSON 文件 (JavaScript Object Notation)

语言类应用程序

- 配置文件解析 (.properties)
- CSV 文件 (Comma-Separated Values)
- JSON 文件 (JavaScript Object Notation)

- SQL 引擎 (Structured Query Language)
- TLA$^+$/TLAPS (TPaxos.tla)
- (Java) 字节码解释器
- C/C++ 语言编译器

语言类应用程序

- 配置文件解析 (.properties)
- CSV 文件 (Comma-Separated Values)
- JSON 文件 (JavaScript Object Notation)

- SQL 引擎 (Structured Query Language)
- TLA$^+$/TLAPS (TPaxos.tla)
- (Java) 字节码解释器
- C/C++ 语言编译器

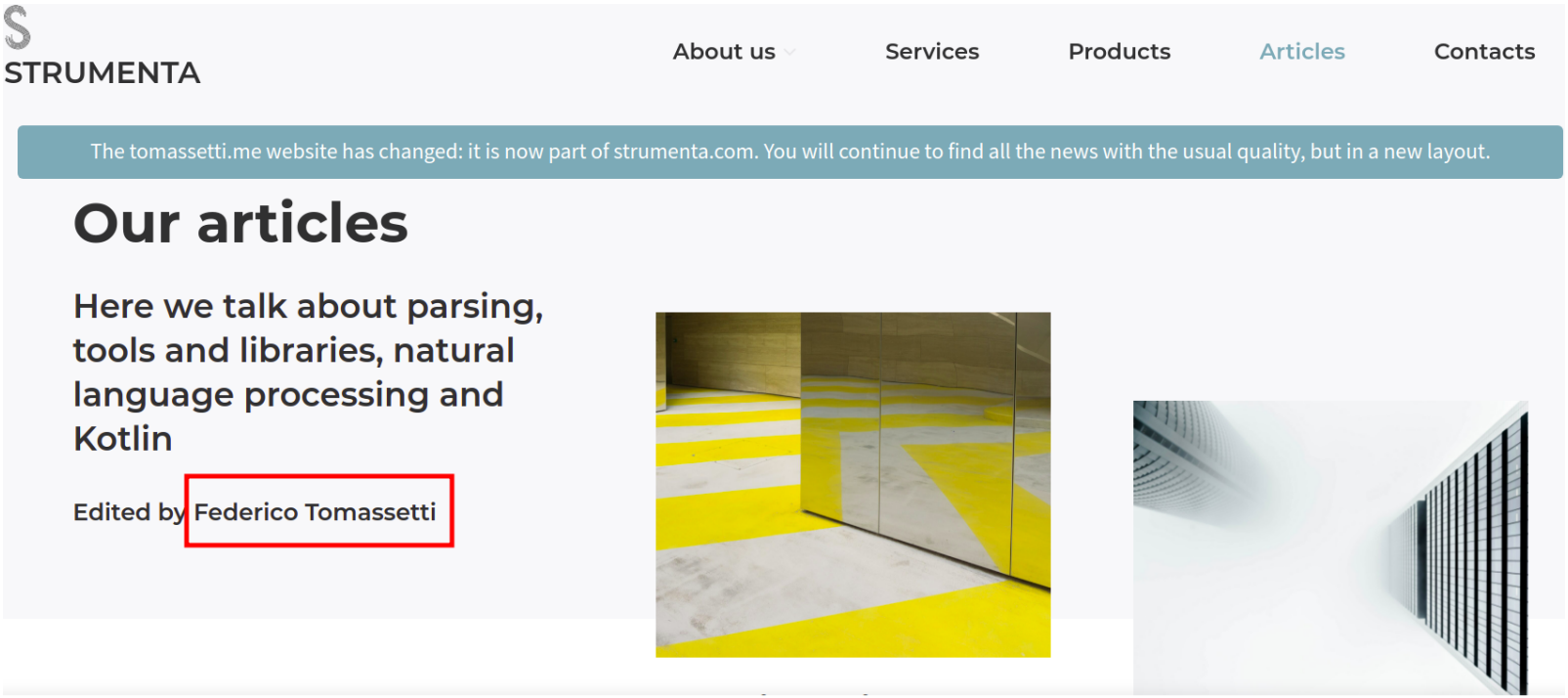
- 排版工具 (LaTeX)
- 绘图工具 (TikZ, Dot/Graphviz)

语言类应用程序

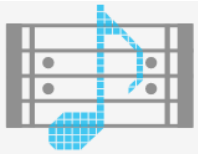
- 配置文件解析 (.properties)
- CSV 文件 (Comma-Separated Values)
- JSON 文件 (JavaScript Object Notation)

- SQL 引擎 (Structured Query Language)
- TLA$^+$/TLAPS (TPaxos.tla)
- (Java) 字节码解释器
- C/C++ 语言编译器

- 排版工具 (LaTeX)
- 绘图工具 (TikZ, Dot/Graphviz)
- L-System (Cantor Set)



tomassetti.me



alda

install tutorial cheat sheet docs community

Alda is a text-based programming language for music composition. It allows you to write and play back music using only a text editor and the command line.

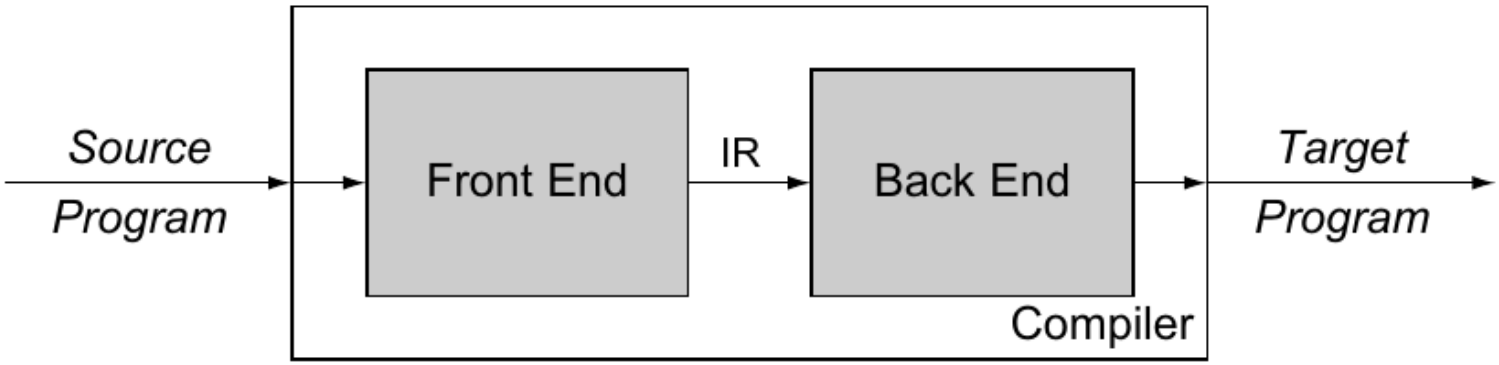
```
piano:
  o3
  g8 a b > c d e f+ g | a b > c d e f+ g4
  g8 f+ e d c < b a g | f+ e d c < b a g4
  << g1/>g/>g/b/>d/g
```

The language's design equally favors aesthetics, flexibility and ease of use.

alda.io

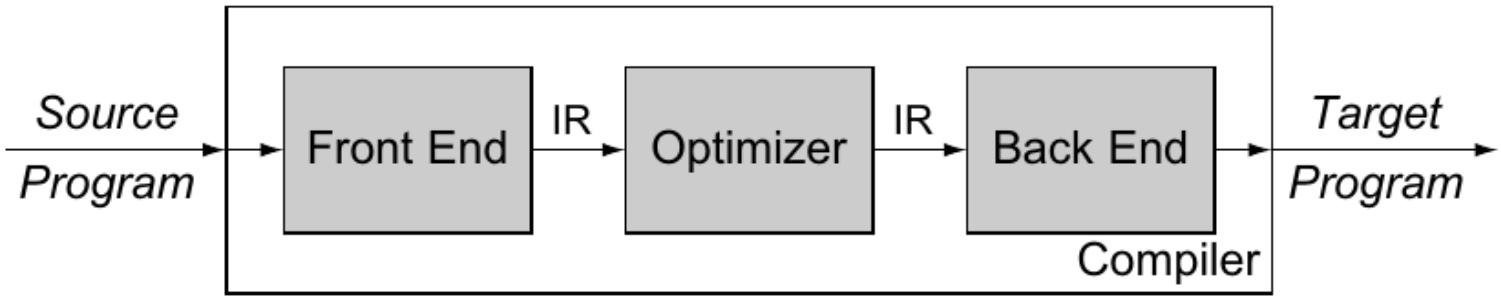
两个月的“编译器设计原理”之旅

IR: Intermediate Representation (中间表示)



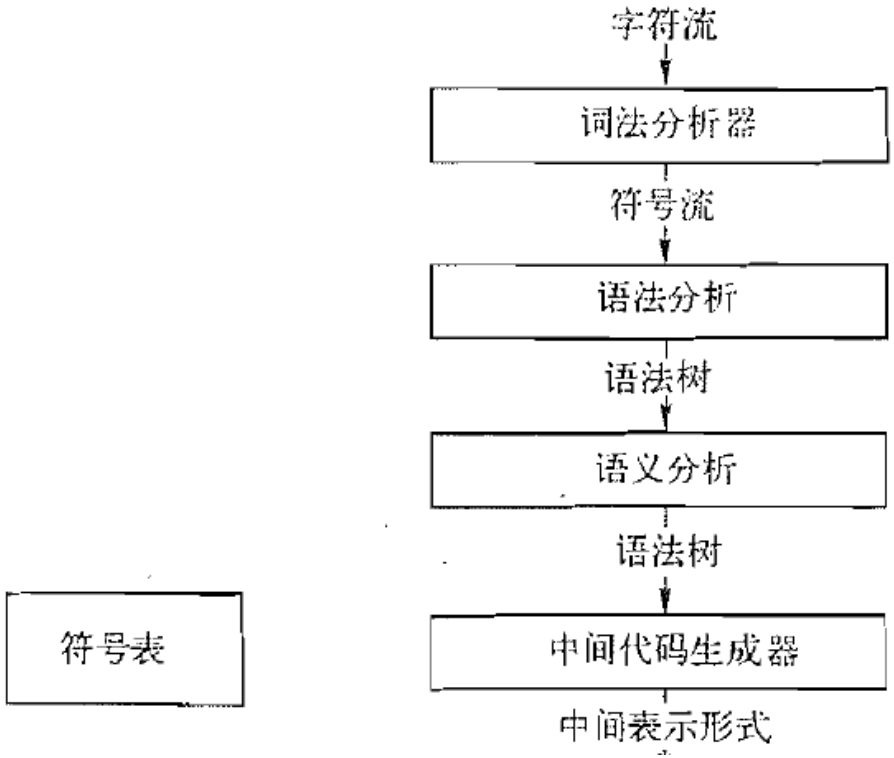
前端 **(分析阶段)**: 分析源语言程序, 收集所有必要的信息

后端 **(综合阶段)**: 利用收集到的信息, 生成目标语言程序



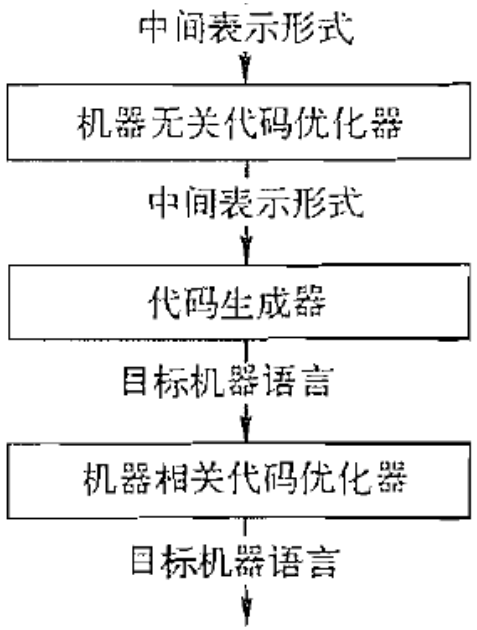
机器无关的中间表示优化

编译器前端：分析阶段



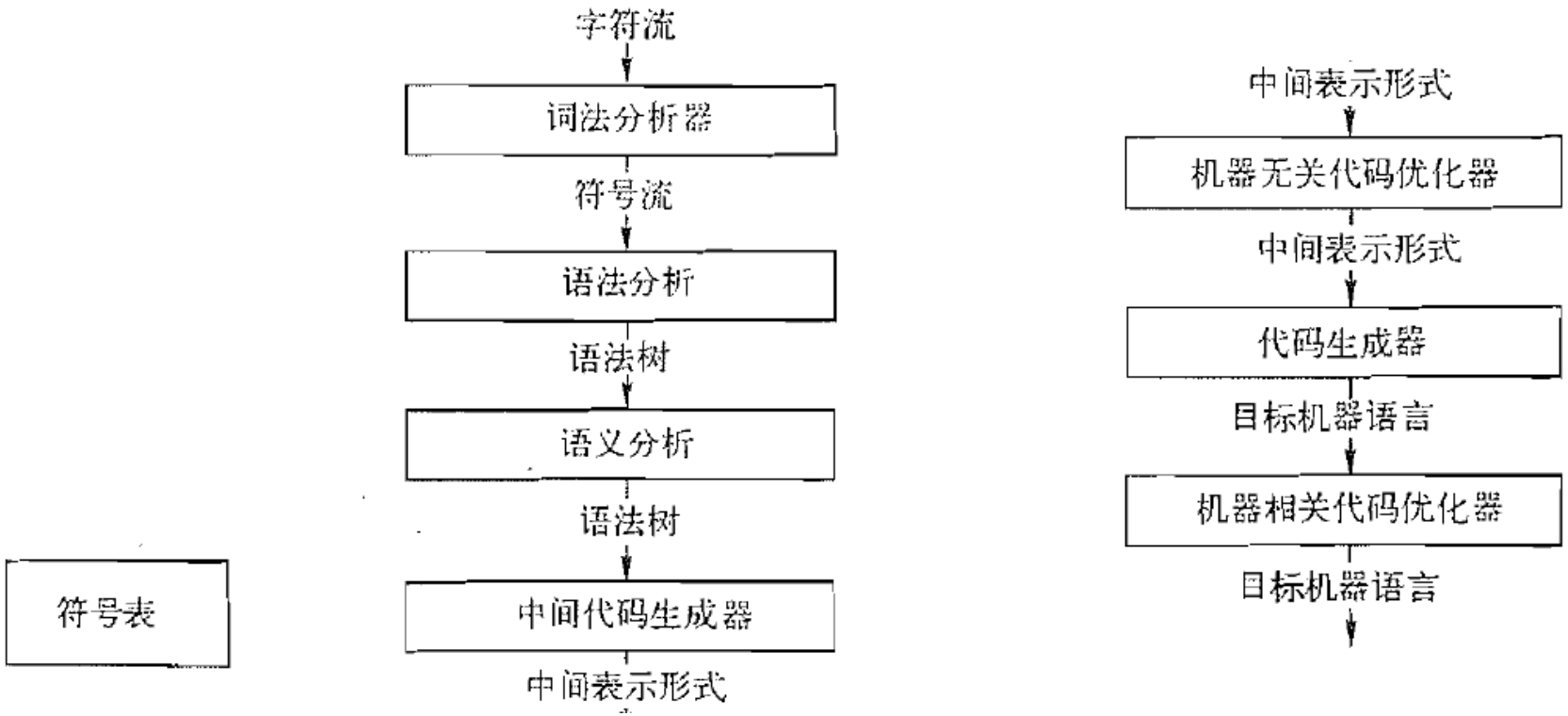
前四次必做实验

编译器后端：综合阶段



第五次选做实验

时间苦短，来不及优化



在设计实际生产环境中的编译器时，**优化**通常占用了大多数时间

Maple IR 及其各种优化技术

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

词法: 标识符、数字、运算符

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

词法: 标识符、数字、运算符

语法: 包含算术运算的赋值语句

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

词法: 标识符、数字、运算符

语法: 包含算术运算的赋值语句

语义: `position`, `initial`, `rate` 是数值类型

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

词法: 标识符、数字、运算符

语法: 包含算术运算的赋值语句

语义: `position`, `initial`, `rate` 是数值类型

物理定律: 当前位置 = 初始位置 + 速度 × 时间

```
position = initial + rate * 60
```

作为一名**程序员**, 你看到了什么?

词法: 标识符、数字、运算符

语法: 包含算术运算的赋值语句

语义: `position`, `initial`, `rate` 是数值类型

物理定律: 当前位置 = 初始位置 + 速度 × 时间

但是, 作为**编译器**, 它仅仅看到了一个**字符串**

词法分析器 (Lexer/Scanner): 将**字符**流转化为**词法单元** (token) 流。

$$\boxed{\text{token} : \langle \text{token-class}, \text{attribute-value} \rangle}$$

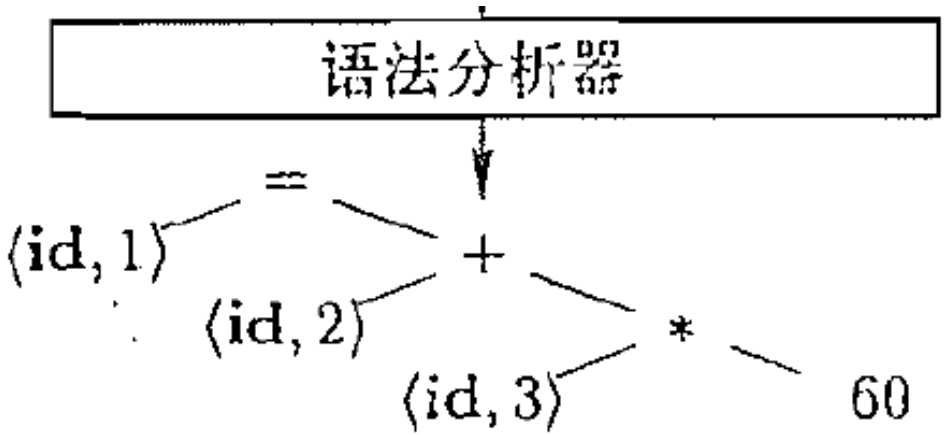
```
position = initial + rate * 60
```

$$\langle \mathbf{id}, 1 \rangle \quad \langle \mathbf{ws} \rangle \quad \langle \mathbf{assign} \rangle \quad \langle \mathbf{ws} \rangle \quad \langle \mathbf{id}, 2 \rangle \quad \langle \mathbf{ws} \rangle$$
$$\langle + \rangle \quad \langle \mathbf{ws} \rangle \quad \langle \mathbf{id}, 3 \rangle \quad \langle \mathbf{ws} \rangle \quad \langle * \rangle \quad \langle \mathbf{ws} \rangle \quad \langle \mathbf{num}, 4 \rangle$$

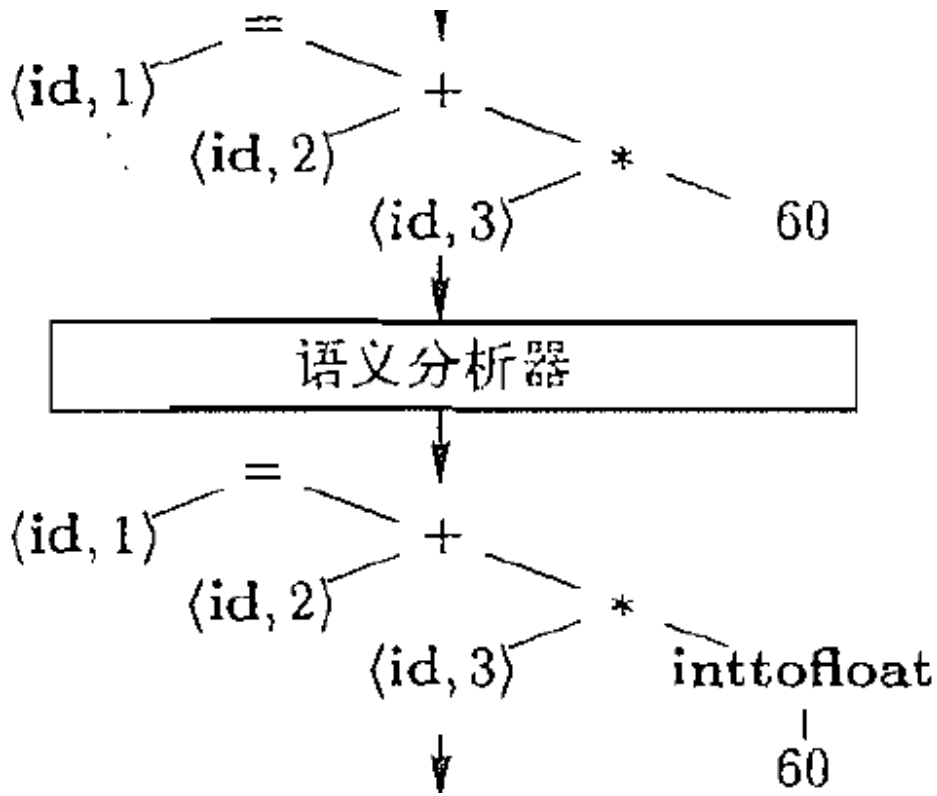
(此处, $1, 2, 3, 4$ 是指向**符号表**的指针)

语法分析器 (Parser): 构建**词法单元**之间的语法结构, 生成**语法树**

```
position = initial + rate * 60
```

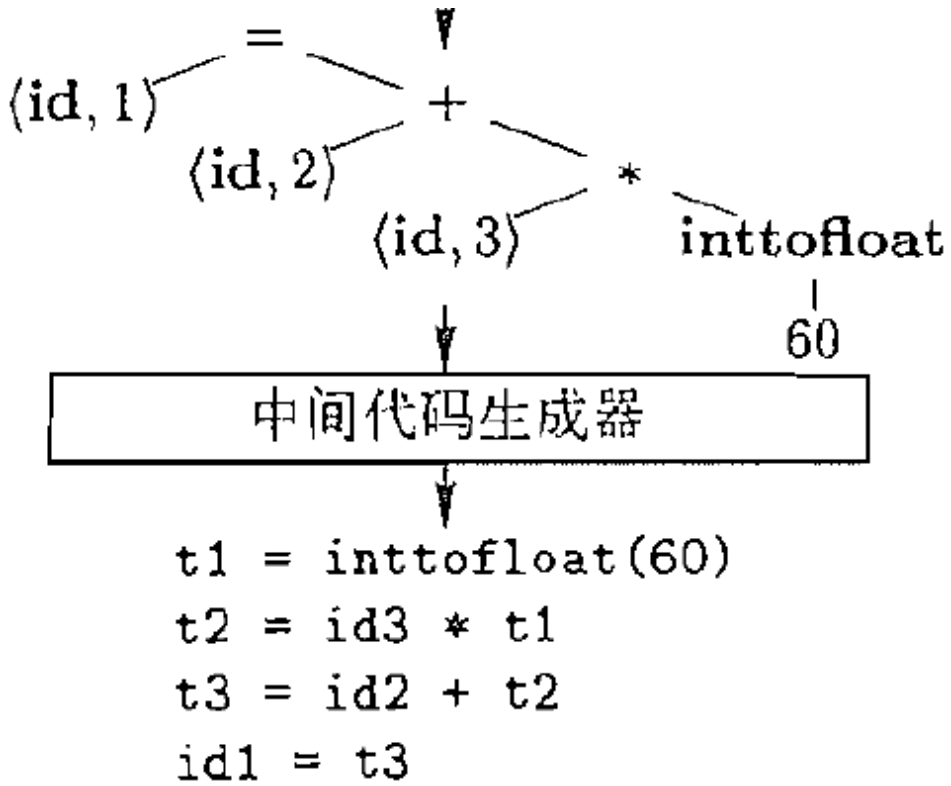


语义分析器: 语义检查, 如**类型检查、“先声明后使用”约束检查**



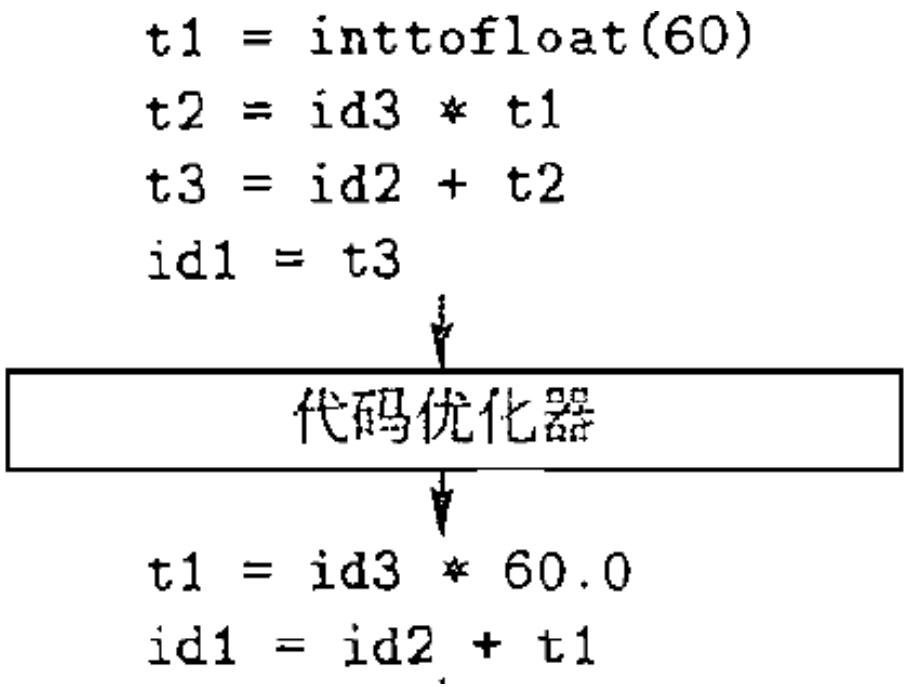
通过语法树上的遍历来完成

中间代码生成器: 生成中间代码, 如 “**三地址代码**”



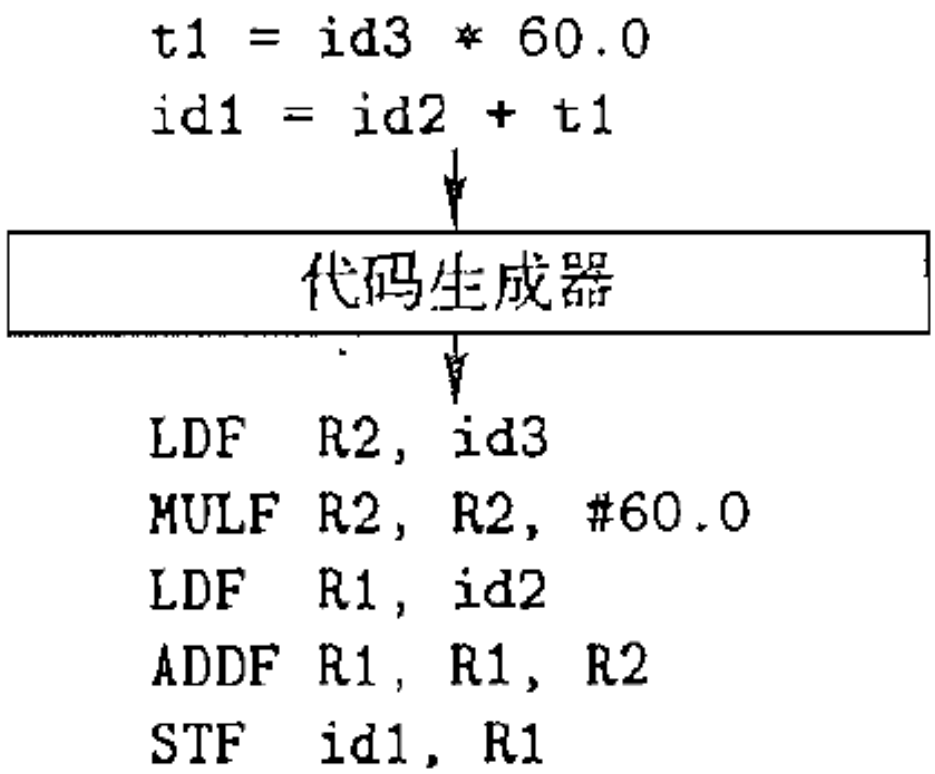
中间代码类似目标代码, 但不含有机器相关信息 (如寄存器、指令格式)

中间代码优化器



编译时计算、消除冗余临时变量

代码生成器：生成目标代码，主要任务包括**指令选择、寄存器分配**



符号表: 收集并管理**变量名/函数名**相关的信息

变量名:

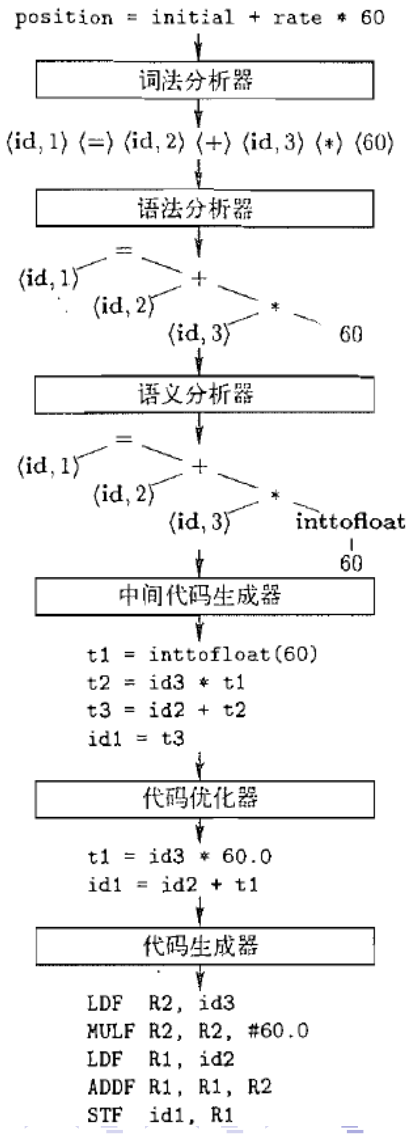
类型、寄存器、内存地址、行号

函数名:

参数个数、参数类型、返回值类型

1	position	···
2	initial	···
3	rate	···

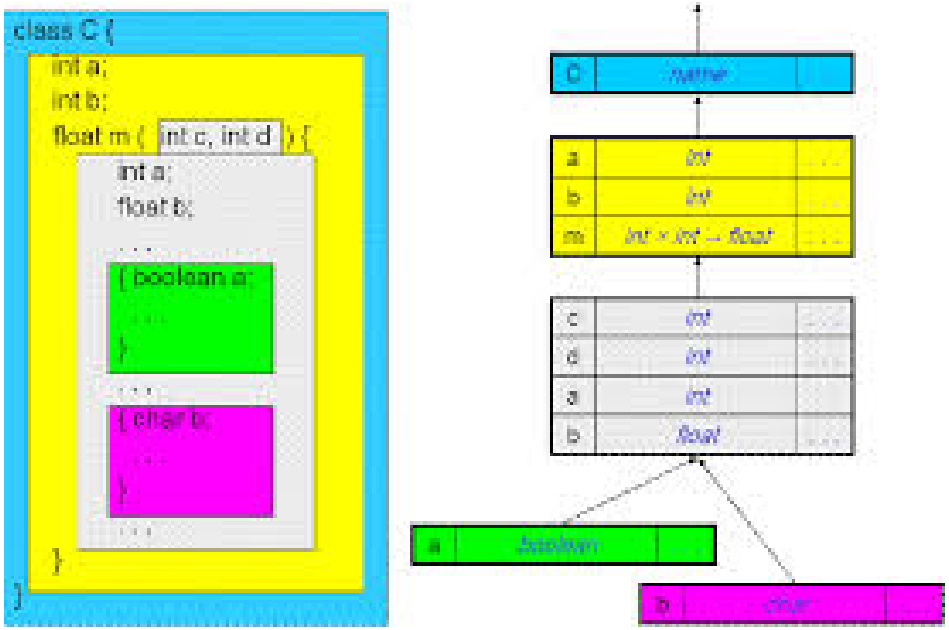
符号表



public class ST<Key extends Comparable<Key>, Value>		
	ST()	*create an empty symbol table*
void	put(Key key, Value val)	*associate* `val` *with* `key`
Value	get(Key key)	*value associated with* `key`
void	remove(Key key)	*remove* `key` *(and its associated value)*
boolean	contains(Key key)	*is there a value associated with* `key`*?*
int	size()	*number of key–value pairs*
Iterable<Key>	keys()	*all keys in the symbol table*

红黑树 (RB-Tree)、哈希表 (Hashtable)

为了方便表达**嵌套结构与作用域**，可能需要维护多个符号表




TIME FOR ACTION

Hello.g4

SimpleExpr.g4

Expr.g4

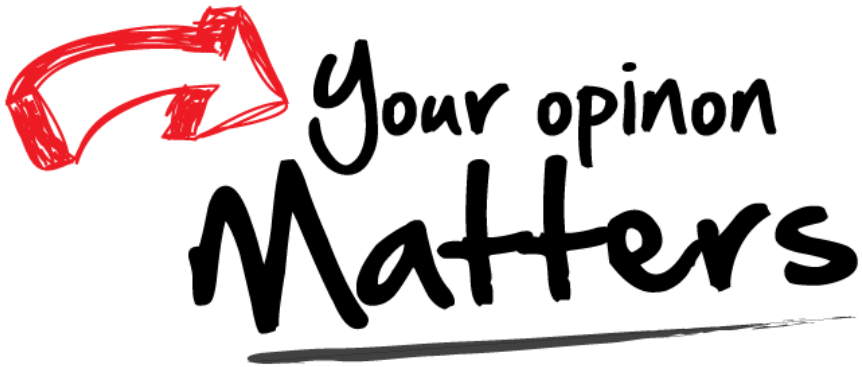
Hello.g4

SimpleExpr.g4

Expr.g4

grammars-v4 @ github


Thank
You!



Office 926

hfwei@nju.edu.cn