

Python 语言程序设计:Python 概述

李宽

likuan@dgut.edu.cn

东莞理工学院

2019.9



目录

- 1 课程简介
- 2 编程语言概览
- 3 程序设计基本方法入门
- 4 Python 编程入门

目录

1 课程简介

2 编程语言概览

3 程序设计基本方法入门

4 Python 编程入门

Python 语言程序设计



意为蟒蛇，拥有者是 PSF(Python Software Foundation, <https://www.python.org>, 非盈利性组织)



Python 语言创立者 Guido van Rossum, 2002: Python 2.x, 2008: Python 3.x

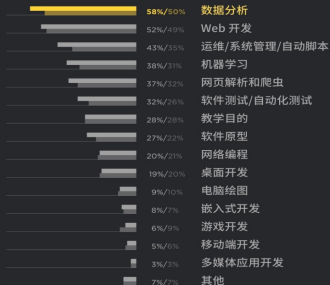
Python 是一门跨平台、开源、免费的解释型高级动态编程语言。

- 谁在用python?
- Facebook, Google, Youtube, Spotify, Netflix, Quora, Dropbox, ...
- 每个AI公司都在用python
- https://www.hartmannsoftware.com/Blog//Companies_Using_Python

Oct 2018	Oct 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.801%	+5.37%
2	2		C	15.378%	+7.00%
3	3		C++	7.593%	+2.59%
4	5	▲	Python	7.156%	+3.35%
5	8	▲	Visual Basic .NET	5.864%	+3.15%
6	4	▼	C#	3.485%	-0.37%
7	7		PHP	2.794%	+0.00%
8	6	▼	JavaScript	2.280%	-0.73%
9	-	▲	SQL	2.038%	+2.04%
10	16	▲	Swift	1.500%	-0.17%
11	13	▲	MATLAB	1.317%	-0.56%
12	20	▲	Go	1.253%	-0.10%

I 你用 PYTHON 做什么

- 2018
- 2017



2017 Average Developer Salary in the U.S.

indeed.com
estimations (USD)

Language

#1 117,147	Ruby/Ruby on Rails
#2 116,027	Python
#3 115,597	C++
#4 115,273	iOS
#5 110,062	JavaScript
#6 102,043	Java
#7 95,045	C
#8 86,354	PHP
#9 85,812	SQL

codementor

- 美国计算机协会会刊（CACM）2014年7月发表的调查报告显示，Python “已经成为目前美国顶尖大学里最受欢迎的计算机编程入门语言”。
- 在计算机专业排名前10的学校里，有8所学校用Python作为编程入门语言
- 在计算机专业排名前39的学校里，有27所学校用Python作为编程入门语言
- Python有望在2019年成为法国高中编程教学语言
- 浙江，北京和山东已经确定把Python编程基础纳入信息技术课程和高考的内容体系
- Python语言已经进入全国计算机等级考试， 2018

课程的定位和目标

课程定位：

- 编程零基础
- 整套 Python 使用（语法 + 扩展库）
- 实践导向

目标：

- 初步掌握抽象思维模式，使用 Python 解决基本问题
- 科研及业界开发常用的扩展库，具备自我提升能力
- 享受编程乐趣！技能+思维
- 为后续算法和数据结构课程学习打下基础

课程安排

时间安排：

- 4 周-13 周，周一 5-7 节
- 4 周-13 周，周三 7-8 节

课程形式：

- 教材：
Python 程序设计基础（第 2 版），董付国等编著，清华大学出版社
- 在线材料，课程网站：<http://144.34.186.36>
- 编程作业 (Educoder.net) + 随堂测验 (雨课堂)

网络资源：

- 廖雪峰的 Python 教程
- <https://github.com/vinta/awesomepython>
- <https://www.practicepython.org/>

MOOC: <https://www.icourse163.org/course/BIT-268001>

考核：期末考试 (闭卷) 50% + 随堂测验 20% + 编程作业 30%

目录

1 课程简介

2 编程语言概览

3 程序设计基本方法入门

4 Python 编程入门

- 什么是计算机？
- 什么是操作系统？
- 什么是编程语言？

语言是什么



语言，广义而言，是用于**沟通**的一套方式，有其**符号**与处理规则，一般称为**文法**。符号通常称为**文字**，会以视觉、**声音**或者**触觉**方式来进行传递。语言用来传递已知或未知事物的含义。“语言”一词可以以更广义的理解为已知或未知世界的基础构成系统。

严格来说，语言是指人类沟通所使用的语言——**自然语言**。在一个先进的社会中一般人都必须通过**学习**才能获得语言能力。语言的目的是交流观念、意见、思想等。**语言学**就是从人类研究语言分类与规则而发展出来的。研究语言的专家被称呼为**语言学家**。

当人发现了某些动物如**海豚**能够以某种方式沟通，就诞生了动物语言的概念。20世纪由于**电脑**诞生，人需要给电脑指令。这种对机器的“单向沟通”就成**电脑语言**。

来源:wikipedia

编程语言是什么



程序员通过编程语言向计算机下达命令
不同的语言可类比不同的语种

Q：计算机^a如何“懂”如此多的语言；

A：“翻译”^b

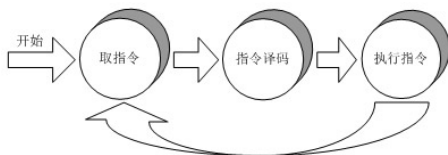
^aCPU 执行的是机器语言

^b编译和解释，后续展开

计算机懂什么语言 1/2

计算机是根据指令操作数据的设备。

- **功能性：**
对数据的操作，表现为数据计算，输入输出处理和结果存储等
- **可编程性：**
根据一系列指令自动地，可预测地，准确地完成操作者的意图



计算机的指令执行单元是 CPU。

CPU 基本工作是执行存储的指令序列¹，即程序。

程序的执行实际上是不断地取出指令、分析指令、执行指令的过程。

¹二进制代码

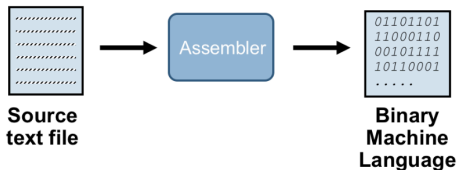
机器语言

用二进制代码表示的计算机能直接识别和执行的机器指令系统的集合。

机器语言 (指令集) 在 CPU 设计制造时就已确定。
不同架构的 CPU 指令集不同, 比如 X86, ARM 系列。
早期使用机器语言直接编程 (拨动开关, 打卡)。

- 优点: 程序占用内存少、执行效率高
- 缺点:
 - 编程工作量大, 容易出错
 - 依赖具体的计算机体系结构, 程序的通用性、移植性差





汇编语言是二进制指令的文本形式，与指令是**一一对应**的关系（助记）

举例：加法指令 00000011 vs. ADD。

只要还原成二进制，汇编语言就可被 CPU 直接执行。

与高级语言比较：

- 优点：高速度和高效率²
- 缺点：机器相关性、编写和调试复杂

²取决于编程者的水平

高级语言包括很多编程语言，如 Java、C、C++、Python 等。

优点:

- 与计算机的硬件结构和指令系统无关
- 有更强的表达能力，可方便地表示数据的运算和程序的控制结构
- 抽象层次更高，能更好的描述各种算法
- 容易学习掌握

缺点: 高级语言编译生成的程序代码**一般**比用汇编程序语言设计的程序代码长(代码体积大)，执行速度慢。

高级语言，汇编，机器语言直观比较

Assembly



C++



Python



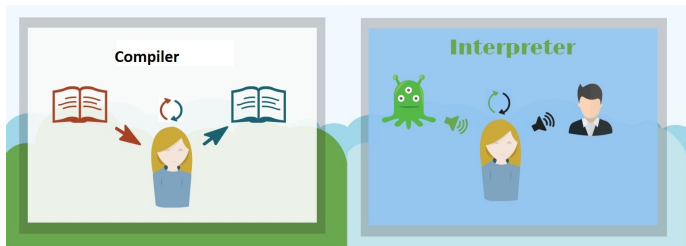
C



高级语言，汇编，机器语言直观比较

code.c	code.s	code.o
1 int accum = 0;	1 .section __TEXT,__text,regular,pure_instructions	1 cffa edfe 0700 0001 0300 0000 0100 0000
2	2 .macosx_version_min 10, 11	2 0400 0000 0002 0000 0020 0000 0000 0000
3 int sum(int x, int y)	3 .globl _sum	3 1900 0000 8801 0000 0000 0000 0000 0000
4 {	4 .align 4, 0x90	4 0000 0000 0000 0000 0000 0000 0000 0000
5 int t = x + y;	5 _sum: ## @sum	5 8c00 0000 0000 0000 2002 0000 0000 0000
6 accum += t;	6 .cfi_startproc	6 8800 0000 0000 0000 0700 0000 0700 0000
7 return t;	7 ## BB@0:	7 0400 0000 0000 0000 5f5f 7465 7874 0000
8 }	8 pushq %rbp	8 0000 0000 0000 0000 5f5f 5445 5854 0000
	9 Ltmp0:	9 0000 0000 0000 0000 0000 0000 0000 0000
	10 .cfi_def_cfa_offset 16	10 2700 0000 0000 0000 2002 0000 0400 0000
	11 Ltmp1:	11 a802 0000 0200 0000 0004 0080 0000 0000
	12 .cfi_offset %rbp, -16	12 0000 0000 0000 0000 5f5f 636f 6d6d 6f6e
	13 movq %rsp, %rbp	13 0000 0000 0000 0000 5f5f 4441 5441 0000
	14 Ltmp2:	14 0000 0000 0000 0000 8800 0000 0000 0000
	15 .cfi_def_cfa_register %rbp	15 0400 0000 0000 0000 0000 0000 0200 0000
	16 movl %edi, -4(%rbp)	16 0000 0000 0000 0000 0100 0000 0000 0000
	17 movl %esi, -8(%rbp)	17 0000 0000 0000 0000 5f5f 636f 6d70 6163
	18 movl -4(%rbp), %esi	18 745f 756e 7769 6e64 5f5f 4c44 0000 0000
	19 addl -8(%rbp), %esi	19 0000 0000 0000 0000 2800 0000 0000 0000
	20 movl %esi, -12(%rbp)	20 2000 0000 0000 0000 4802 0000 0300 0000
	21 movl -12(%rbp), %esi	21 b582 0000 0100 0000 0000 0002 0000 0000
	22 addl _accum(%rip), %esi	22 0000 0000 0000 0000 5f5f 656b 5f6b 7261
	23 movl %esi, _accum(%rip)	23 6d65 0000 0000 0000 5f5f 5445 5854 0000
	24 movl -12(%rbp), %eax	24 0000 0000 0000 0000 4800 0000 0000 0000
	25 popq %rbp	25 4000 0000 0000 0000 6802 0000 0300 0000
	26 retq	26 0000 0000 0000 0000 0b00 0068 0000 0000
	27 .cfi_endproc	27 0000 0000 0000 0000 2400 0000 1800 0000
	28	28 000b 8a00 0000 0000 0200 0000 1800 0000
	29 .globl _accum ## @accum	29 c002 0000 0200 0000 c002 0000 1800 0000
	30 .zerofill __DATA,__common,_accum,4,2	30 0b00 0000 5000 0000 0000 0000 0000 0000
	31	31 0000 0000 0200 0000 0200 0000 0000 0000
	32 .subsections_via_symbols	32 0000 0000 0000 0000 0000 0000 0000 0000
	33	33 0000 0000 0000 0000 0000 0000 0000 0000
		34 0000 0000 0000 0000 0000 0000 0000 0000
		35 5548 89e5 897d fc89 75f8 8b75 fc03 75f8
		36 8975 f48b 75f4 0335 0000 0000 8935 0000
		37 0000 8b45 f45d c300 0000 0000 0000 0000
		38 2700 0000 0000 0001 0000 0000 0000 0000
		39 0000 0000 0000 0000 1400 0000 0000 0000
		40 017a 5200 0178 1001 100c 0708 9001 0000
		41 2400 0000 1c00 0000 98ff ffff ffff ffff
		42 2700 0000 0000 0000 0041 0e10 8602 430d
		43 0600 0000 0000 0000 1e00 0000 0000 001d
		44 1800 0000 0000 001d 0000 0000 0100 0006
		45 0600 0000 0f02 0000 8800 0000 0000 0000
		46 0100 0000 0f01 0000 0000 0000 0000 0000
		47 005f 7375 6d00 5f61 6363 756d 0000 0000
		48

高级语言: 编译型与解释型 1/2



- 编译器:
将源代码转换成其他的更低级代码 (如二进制码), 但不会执行
- 解释器:
读取源代码, 生成指令让计算机硬件执行, 不会输出另外一种代码

类比: 翻译成文件 vs 同声传译

性能 Performance

● 编译器

- 事先用较多时间把整个程序的源代码编译成另外一种代码
- 执行的效率**往往**更高
- 时间消耗在预编译的过程中

● 解释器

- 逐行读取源代码，解释，然后立即执行
- 往往使用相对简单的词法/语法分析，生成机器码，交由硬件执行
- 提高效率的关键：减少解释的次数和复杂性以压缩解释时间

编译器和解释器的界限在模糊：对 Python 而言，支持伪编译为字节码，支持 pyinstaller 等工具打包形成完整的可执行文件。

高级语言: 动态类型和静态类型

动态类型语言：

- 运行期间才去做数据类型检查
- 无需给变量指定数据类型
在第一次赋值给变量时，在内部将数据类型记录下来

静态类型语言：

- 数据类型是在编译期间检查的
- 写程序时要声明所有变量的数据类型

Revisit: Python 是一门跨平台、开源、免费的解释型高级动态编程语言。

目录

- 1 课程简介
- 2 编程语言概览
- 3 程序设计基本方法入门**
- 4 Python 编程入门

What & How: 从实例讲起 1/3

实例: 求一个正数的平方根

非正式定义:

对于正数 x , 如果有数 $r, r \geq 0$, 且 $r^2 = x$, 称 r 为 x 的平方根。

非正式定义的形式化**声明**:

```
square_root(x){  
    找到一个 $r$ , 确保 $r$  的平方等于 $x$   
    返回 $r$ }
```

理想: 有某种语言可直接用上述声明生成代码 (what)

现实: 程序员必须要告诉计算机到底该怎么做 (How)

- 第一步怎么做
- 第二步怎么做
- ...
- 判定结束

What & How: 从实例讲起 2/3

Write a function `[sqrtx] = sqRoot(x, tol)` that computes the square root of x (1 x 1 double) to a given tolerance `tol` (1 x 1 double). The square root of a positive number can be approximated by the following iterative method:

$$y_{n+1} = \frac{1}{2} \left(y_n + \frac{x}{y_n} \right)$$

where $y_0 = x$ and $y_n \rightarrow \sqrt{x}$ as n approaches ∞ . In your function, you want to use a while loop to iterate until the absolute value of $((y_{n+1})^2 - x)$ is less than or equal to the tolerance given by the variable `tol`. i.e. $|y_{n+1}^2 - x| \leq \text{tol}$.

牛顿法

- **step1** 先猜测 $r =$ 某初值
- **step2** 判断 r^2 和 x 是否非常接近 (如相差 0.0001)
- **step3** 如果不接近, $r = \frac{r+x/r}{2}$, 继续**step2**; 否则, 输出结果 r

What & How: 从实例讲起 3/3

举例: 求 2 的平方根

猜测值(r)	r的平方和x 的差值	商 (x/r)	平均值
1	1	$2/1 = 2$	$(2+1)/2 = 1.5$
1.5	0.25	$2/1.5 = 1.3333$	$(1.5+1.3333)/2 = 1.4167$
1.4167	0.007	$2/1.4167 = 1.4118$	$(1.4167+1.4118)/2 = 1.4142$
1.4142	-0.00003

```
square_root(x){  
    r=1;  
    while(abs(r*r-x)>0.0001){  
        r=(r+x/r)/2;  
    }  
    return r;  
}
```

用户描述的需求 (通常是声明式的), “做什么”



计算机可以理解, 可以运算的步骤, “怎么做”

程序的基本编写方法:IPO

IPO:

- I: Input, 程序的输入 (文件, 网络, 控制台, 交互界面...)
- P: Process, 对输入数据进行计算产生输出结果的过程, 主要逻辑
- O: Output, 程序展示运算结果 (图形, 文件, 网络等等...)

6 个步骤

- 分析问题, 明确问题的计算部分
- 划分边界, 确定问题的功能边界, 规划 IPO
- 设计算法, 设计问题的求解算法
- 编写程序, 用编程语言正确实现
- 调试测试, 排错检查
- 升级维护, 根据需求升级调整

为什么要学习计算机编程

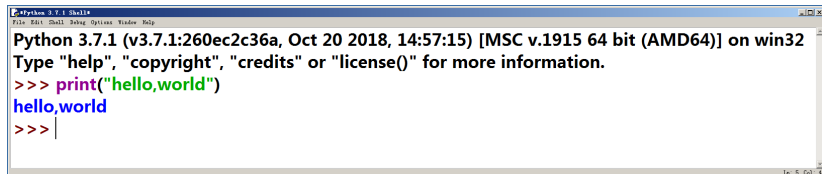
- **训练思维**: 抽象交互关系, 自动化执行的思维模式 (计算思维)
- **增进认识**: 求解问题之外, 还要考虑用户体验, 执行效率等
- **享受乐趣**: 展示自我, 提升心理满足感, 创新创业
- **提高效率**: 解决工作中的实际问题
- **职业前景**: 科研或就业都必须



目录

- 1 课程简介
- 2 编程语言概览
- 3 程序设计基本方法入门
- 4 Python 编程入门

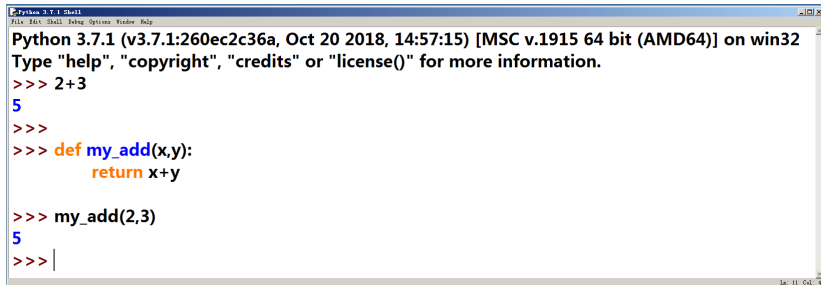
Python 直观印象 1/3



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("hello,world")
hello,world
>>> |
```

hello,world

关键词：命令 (How) vs 函数 (What)



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 2+3
5
>>>
>>> def my_add(x,y):
        return x+y

>>> my_add(2,3)
5
>>> |
```

关键词：**命令式** vs **文件式**，在 Windows 下 Python 开发环境安装中讨论

关键词：**胶水语言**

能够把用其他语言编写的各种模块（尤其是 C/C++）无缝拼接，更好地发挥不同语言和工具的优势，满足不同应用领域的需求。

Python 开发环境



www.IndianAIExpert.com



& Many more



Python 版本之争



Python 目前存在 2.x 和 3.x 两个系列的版本，互相之间**不兼容**。

Python 2.x 系列最迟将于 2020 年全面放弃维护和更新。

本课程选用最新版本**Python 3.7.X³**进行教学。

建议：掌握 Python 语言使用思路，自我学习，灵活切换⁴

³<https://www.python.org> released

⁴可自行网上查阅 2.x 和 3.x 版本之间的差异

Windows 下 Python 开发环境安装

课程主页:<[install_python3.7_for_windows.pdf](#)>

问题：提示用户输入，根据标识 (F: 华氏温度,C: 摄氏度) 判断输入的是摄氏度还是华氏度，根据转换规则完成计算

- ① 分析问题的计算部分
- ② 确定功能, 使用 IPO 方法进一步分析
 - 输入：华氏或者摄氏温度值、温度标识
 - 处理：温度转化算法
 - 输出：摄氏或者华氏温度值、温度标识
- ③ 设计算法, $C = (F - 32) / 1.8$, $F = C * 1.8 + 32$
- ④ 编写程序
- ⑤ 调试、运行程序
- ⑥ 升级和维护

Python 实例, 温度转换 2/2

```
1  #TempConv.py
2
3  val = input(" 请输入带温度表示符号的温度值 (例如: 32C): ")
4
5  if val[-1] in ['C','c']:
6      f = 1.8 * float(val[0:-1]) + 32
7      print(" 转换后的温度为: %.2fF"%f)
8  elif val[-1] in ['F','f']:
9      c = (float(val[0:-1]) - 32) / 1.8
10     print(" 转换后的温度为: %.2fC"%c)
11 else:
12     print(" 输入有误")
```

Python：严格缩进

后续围绕此示例展开讲解 Python 语法元素

Python 标准库

python 标准库是用 python 和 C 语言预先编写的模块，随 python 解释器一起自动安装。python 标准库本身虽然不属于核心语言，却是 python 系统的标准组成部分。

举例：利用标准库 math 计算给定弧度值的余弦：

```
1  # 计算余弦
```

```
2  
3  from math import cos
```

```
4  
5  val = input(" 请输入弧度值: ")
```

```
6  c = cos(float(val))
```

```
7  print(" 对应的余弦值为: %.2f"% c)
```

Python 第三方库

实际开发中常需要单独获取和安装一些第三方库。
第三方库很多是对标准库的优化和再封装。

举例：利用第三方库 `statistics` 计算一组给定数的平均值：

```
1  # 计算平均数
2
3  import statistics
4
5  v=statistics.mean([1,2,3,4,5,6,7,8,9])
6  print("1-9 的平均值为: %.2f"% v)
```

第三方库的安装（命令行环境下，`pip install 模块名`⁵）

⁵如果没有 `pip`，执行命令 `python -m ensurepip --default-pip`

本课作业：

- ① 根据课程主页 Word 文档说明下载并安装 Python3.7.X 开发环境
- ② 在课程主页上下载本节课对应的 3 个 py 文件 (TempConv.py, CalcCos.py, CalcMean.py)，并运行成功
- ③ 使用 `pip install statistics` 在本机上安装 statistics 第三方库，改写 CalcMean.py 使其计算一组数的样本方差