



School of Economics and Management, Beihang University

# 现代程序设计技术

赵吉昌

[jichang@buaa.edu.cn](mailto:jichang@buaa.edu.cn)

[www.zhaojichang.cn](http://www.zhaojichang.cn)

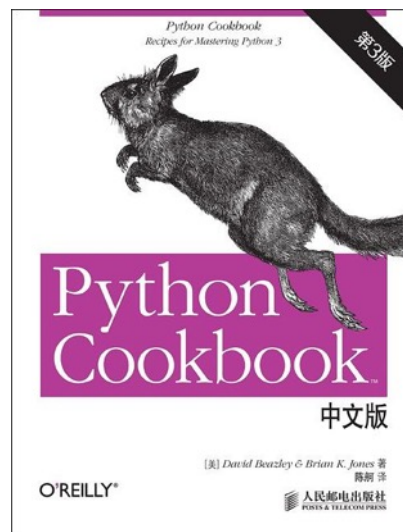
- 答疑方式
  - 课前课后：一般会早到15分钟左右
  - 办公室：新主楼A1027（周一、三、周四上午8：00到晚上21：00）
  - 邮件：[jichang@buaa.edu.cn](mailto:jichang@buaa.edu.cn)（尽快回复）
  - 手机：13910909034（急事）

- 什么叫现代程序设计技术
  - 怎么样就是现代
  - 怎么样就叫设计
- 现代程序设计技术包括哪些
  - **面向对象**
  - **设计模式**
  - 项目管理
  - 团队合作
  - 软件工程方法
  - 软件测试

- 这门课的特点
  - 讲授语言
    - 不涉及太多语法细节
    - 在对Python的基本掌握基础上进一步涉及**更丰富的编程场景**
  - 更强调实际问题
    - 数理基础更加扎实
    - 问题规模和难度可以相应提高
  - 课程的目的
    - 熟练掌握面向对象编程并能够利用面向对象编程实现**中小规模**软件
    - **重应用的编程训练**

- **精通**一门面向对象的程序设计语言
  - Python
  - 熟练使用常用的数据结构和功能库
- 掌握中小规模软件的开发
  - 争取写到3000-5000行
  - 多类型、中大规模数据处理
  - 并行并发
  - 有一定的智能化
  - **实际有用**

Python tutorial: <http://www.pythondoc.com/pythontutorial3/index.html>



- 不建议购买相关书籍
  - 一般厚而贵
  - 相关内容极易失效（技术发展过快）
  - 编程库等的说明文档和实现示例网上更新最快
  - 课程中心会上传电子书
- 多上网查资料
  - 程序员们会分享很多心得
  - [csdn](#), [stackoverflow](#)
  - [github](#)（建立自己的帐号以累积代码项目）
- 实际问题驱动
  - 学技术而非上“课”
  - 将有力地支撑所属专业

- 平时成绩
  - 30%
  - 每周有小作业（课程中心）
  - 个人完成
- 期末成绩
  - 70%
  - 判断
  - 选择
  - 编程题（不涉及算法和数据结构，体现面向对象编程和设计思想）
  - 不考记忆力





- 关于考勤
  - 不点名
  - 来不来绝对不会影响成绩
  - 面向对象特别熟悉并精通Python的同学可以不来
- 关于作业
  - 平时小作业按次计分
  - 不接受逾期提交
  - 如果作业有问题会实时干预
  - 会公开优秀作业供其他同学学习

- 平时作业
  - 一共15次，一次2次
  - 按时提交不能延误
  - 每周提交后生成当次成绩，后续不再调整

Not Secure — 202.112.129.14

评价信息

优秀之处	不足之处
老师上课很用心负责，讲解风趣幽默，能极大地提高同学们的能力	期末考试有点过于难了。。。
认真负责	无
吉昌老师真的太好了	暂无
本学期唯一觉得有意思的课了，尤其是作业。在经管众多只要背书就能拿高分的课程中，为数不多的实操课。希望以后能加大力度（不是）。大作业该布置还是要布置的，期末考试u1s1没啥卵用（虽然选填题还是比较有趣的吧），建议每次作业5分，共15次作业。最后来一次大作业25分美滋滋。功夫都花在平时上期末也就不需要准备考试了（虽然这考试也没啥好准备的）	期末占比感觉还是有点高
--	作业和讲课内容差别太大，作业太难！！
课程较难，作业量偏大，但是吉昌老师人超级好，对同学们的作业耐心指导，讲课也很棒	无

Not Secure — 202.112.129.14

评价信息

优秀之处	不足之处
课程内容更新及时，与实践联系紧密。能学到的知识非常多，收获很大。	考试有英文试题未事先通知
责任心强，教学态度好，答疑及时热情！	--
老师非常nice 细致认真 随时改进课堂教学	--

- 由硕士生和博士生担任助教
  - 随机检查每周作业
  - 归纳共性问题
  - 发现优秀示范
- 助教联系方式
  - 杨阳
  - 邮件：[yangyang\\_buaa@buaa.edu.cn](mailto:yangyang_buaa@buaa.edu.cn)
  - 主要负责作业出题，检查和期末的试卷批改
  - 如果有问题我无法解决，尽量周末跟他联系（科研压力太大）

- 带电脑来上课
- 便学便写便运行
- 鼓励自学
- 鼓励解决实际问题
  - 课题研究
  - 工程项目



- 日慎一日，以求其事之济，一怀焦愤之念，则恐无成耳。千万忍耐，千万忍耐！  
“久而敬之”四字，不特处朋友为然，即凡事亦莫不然。
- 试玉要烧三日满，辨材须待七年期

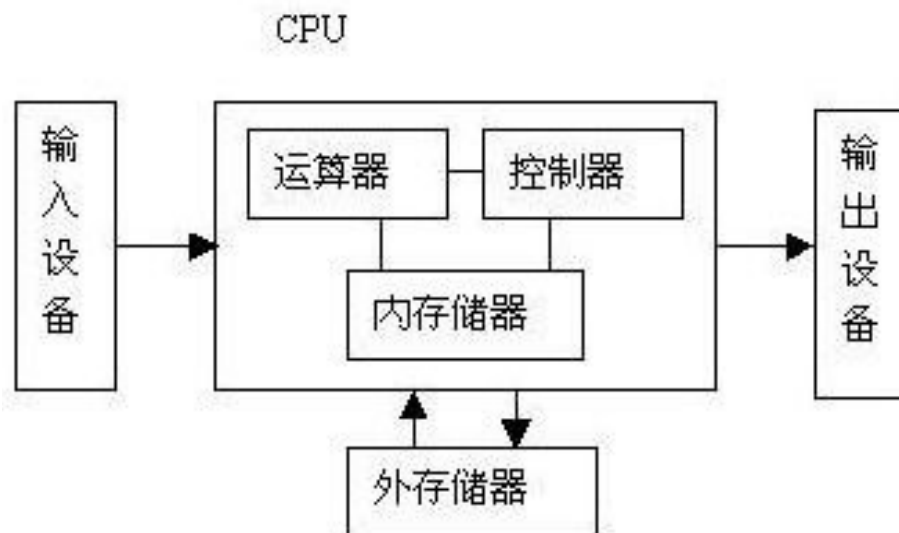


# 什么是生产力？



- 编程回顾
- 面向对象简介
- Python发展简史
- Python开发环境配置

- 依然基于冯诺依曼结构
  - CPU
  - 内存





- 量子计算机等很高大上，但是...
  - 量子计算机之所以快，和量子计算机本身的叠加特性有关，但仅体现在处理特定问题（如检索）上
  - 经典计算机“慢”也可能是因为对一些问题我们还没有找到更好的算法
  - 量子计算机的出现并不代表经典计算机将退出历史舞台
  - 我们学的编程技术还会有用许多年
  - 我们学的设计思想将会有用更多年

这样  
我就放心了...





- 编程回顾
- 面向对象简介
- Python发展简史
- Python开发环境配置

- 面向过程
  - 解决某个问题的流程
  - 1+1
- 结构化设计
  - 功能模块
  - 复用：不再闭门造车
- 数据对象
  - `struct Student {int id; char [] name;;}`
  - 有了初步的封装思想
  - 除了属性，还有行为

- 面向对象
  - object oriented (OO)
  - 将现实世界的事物抽象成对象
  - 包含属性和基于这些属性的行为
  - 对象作为程序的基本单元，将程序和数据封装其中，以提高软件的重用性、灵活性和扩展性
- 基本特点
  - 封装
  - 继承
  - 多态
- Shape示例

- 我们主要学习如下内容
  - Python编程基础
    - 掌握基本的语法规则和库函数使用
  - 类
  - 继承
  - 异常处理
  - 装饰器
  - 抽象类
  - 生成器与迭代器
  - 多进程
  - 多线程
  - 网络编程
  - 异步IO
  - 数据库编程



- 编程回顾
- 面向对象简介
- Python发展简史
- Python开发环境配置

- Python is an **interpreted**, high-level, general-purpose programming language. Created by **Guido van Rossum** and first released in 1991.
- Its language constructs and object-oriented approach aim to help programmers write clear, logical code for **small and large-scale** projects.
- Python is **dynamically typed and garbage-collected**. It supports multiple programming paradigms, including procedural, **object-oriented**, and functional programming.
- Python 2.0, released 2000.
- Python 3.0, released 2008, was a major revision of the language that is **not completely backward-compatible**, and much Python 2 code does not run unmodified on Python 3 (was extended to 2020).
- Guido van Rossum shouldered sole responsibility for the project until July 2018 but now shares his leadership as a member of a five-person steering council.












# 为什么以Python为例



- Oracle收购Sun的影响
- Google对Kotlin语言的推广
- 数据科学的重要性

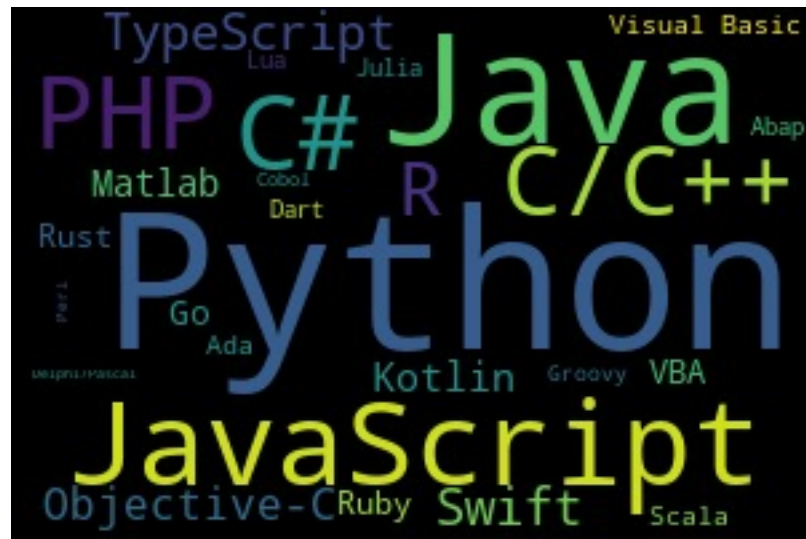
The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings.

Aug 2021	Aug 2020	Change	Programming Language		Ratings	Change
1	1			C	12.57%	-4.41%
2	3			Python	11.86%	+2.17%
3	2	▼		Java	10.43%	-4.00%
4	4			C++	7.36%	+0.52%
5	5			C#	5.14%	+0.46%
6	6			Visual Basic	4.67%	+0.01%
7	7			JavaScript	2.95%	+0.07%

- The PYPL Popularity of Programming Language Index is created by analyzing how often language tutorials are searched on Google. The more a language tutorial is searched, the more popular the language is assumed to be. It is a leading indicator. The raw data comes from Google Trends

Worldwide, Sept 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	29.48 %	-2.4 %
2		Java	17.18 %	+0.7 %
3		JavaScript	9.14 %	+0.8 %
4		C#	6.94 %	+0.6 %
5		PHP	6.49 %	+0.4 %
6		C/C++	6.49 %	+0.9 %
7		R	3.59 %	-0.5 %
8	↑↑↑	TypeScript	2.18 %	+0.3 %
9		Swift	2.1 %	-0.4 %
10	↓↓	Objective-C	2.06 %	-0.6 %
11	↑	Kotlin	1.91 %	+0.3 %

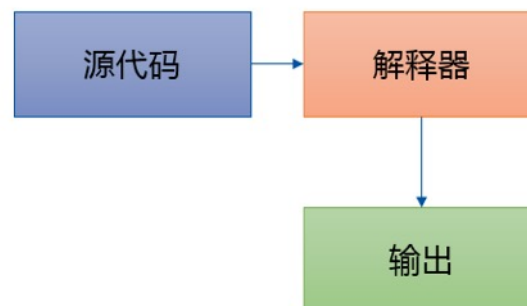
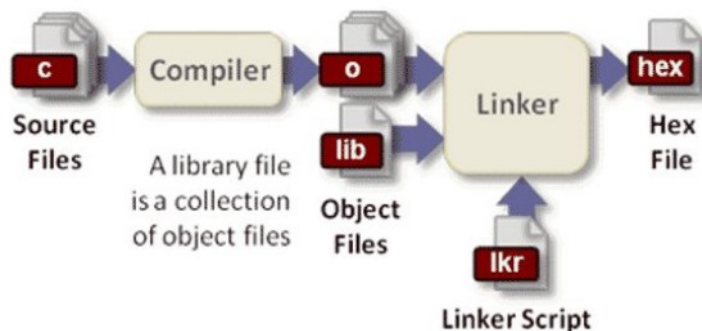


- IEEE Spectrum Rank

Rank	Language	Type	Score
1	Python <sup>▼</sup>	  	100.0
2	Java <sup>▼</sup>	  	95.4
3	C <sup>▼</sup>	  	94.7
4	C++ <sup>▼</sup>	  	92.4
5	JavaScript <sup>▼</sup>		88.1
6	C# <sup>▼</sup>	   	82.4
7	R <sup>▼</sup>		81.7
8	Go <sup>▼</sup>	 	77.7

- 解释执行

- 解释器与编译器有相似，也有区别



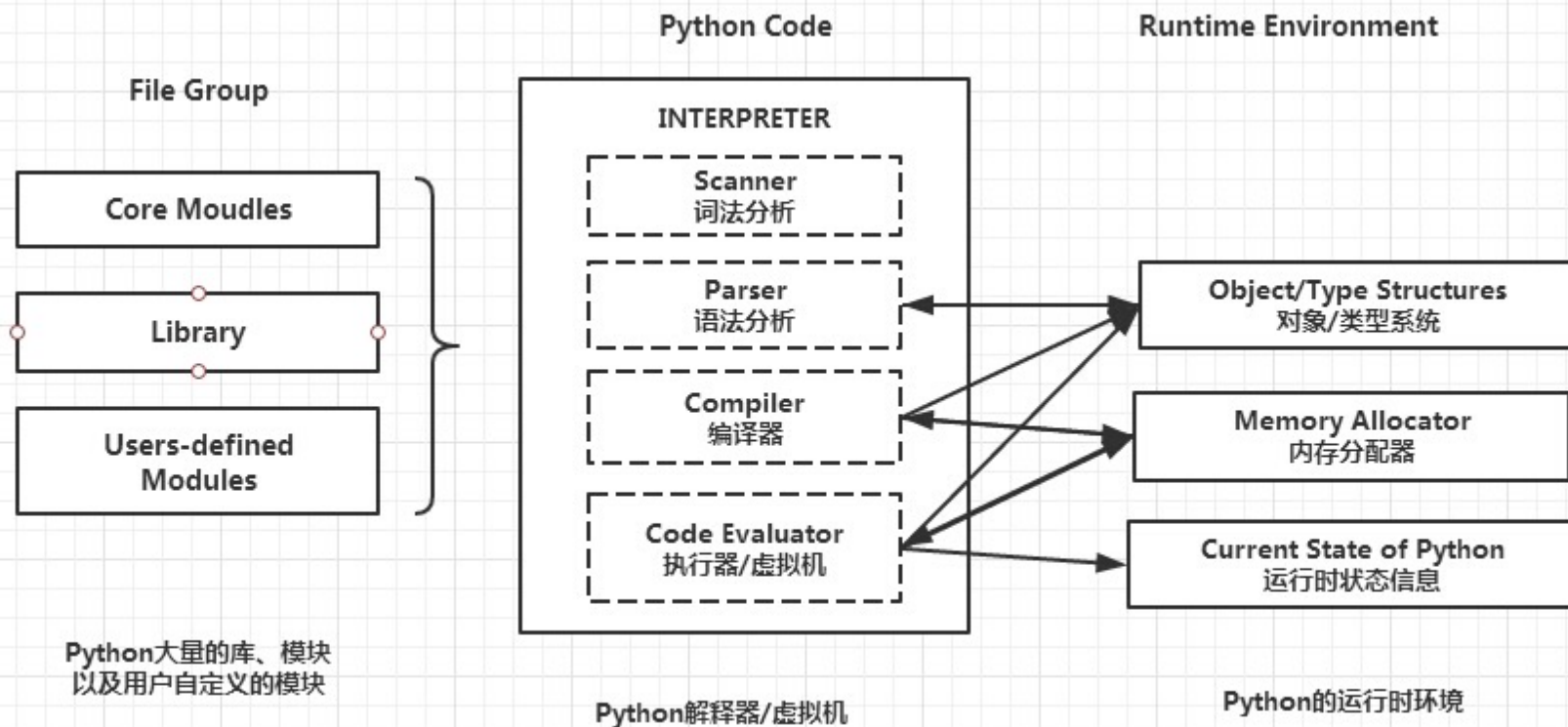
**编译器：先整体编译再执行**

**编译方式：运行速度快，但任何一个小改动都需要整体重新编译。可脱离编译环境运行。代表语言是C语言。**



**解释器：边解释边执行**

**解释方式：运行速度慢，但部分改动不需要整体重新编译。不可脱离解释器环境运行。代表语言是Python语言。**



- CPython：官方版本的解释器，C语言开发，使用最广
  - 注意不是Cython
- IPython：基于CPython之上的一个交互式解释器，在交互方式上有所增强，执行Python代码的功能和CPython是完全一样的
- PyPy：一个追求执行速度的Python解释器。采用JIT技术，对Python代码进行动态编译（注意，不是解释），可以显著提高Python代码的执行速度
- Jython：运行在Java平台上的Python解释器，可以直接把Python代码编译成Java字节码执行
- IronPython：运行在微软.Net平台上的Python解释器，可以直接把Python代码编译成.Net的字节码
- 在执行程序时，解释器逐行读取源代码并逐行解释运行
  - 为了减少这一重复性的解释工作，引入了pyc文件



- 编程回顾
- 面向对象简介
- Python发展简史
- Python开发环境配置

- 版本
  - Python 3
  - 下载并安装对应平台的解释器
- 集成开发环境
  - 需要
  - 项目管理
  - 智能提示（使用大量核心库或第三库时较为方便，也是现代程序设计的一个特点）
  - 代码格式智能整理





- 1. 搭建python开发环境（无需提交）
  - 确定开发IDE
- 2. 配置开发环境（无需提交）
  - 用pip或anaconda管理包
  - 基础功能或技巧：
    - virtualenv：或者直接使用conda的虚拟环境
    - tqdm：进度条，训练或者数据加载非常有用
    - json/demjson：大部分数据以json格式存储，部分不标准的json文本需要利用demjson
    - pickle：结果序列化存储
    - argparse：交互参数解析
  - 数据处理基础：
    - numpy：数组，
    - scipy
    - pandas
  - 可视化：
    - matplotlib：最常用的绘图工具
    - seaborn：辅助matplotlib使用
    - forlium/pyecharts(python中只推荐地图绘图部分，建议利用原生的js配合系统开发库实现功能更多的图)

## • 2. 配置开发环境续

### – 爬虫：

- requests/urllib:发出基本的网络请求
- BeautifulSoup:主要功能是html内容的解析
- Scrapy：基本的爬虫与数据采集
- Selenium：模拟浏览器访问，和Scrapy等配合使用

### – 文本处理：

- jieba：分词工具
- gensim:话题模型及word2vec嵌入等

### – 图/网络数据：

- networkx：复杂网络分析

### – 图片处理初步：

- pillow(PIL):图片的基本变换，深度学习部分要用到图片数据
- opencv-python:图片+视频数据

### – 音频处理初步：

- librosa

### – 统计模型：

- statsmodels

- 2. 配置开发环境续
  - 机器学习：
    - scikit-learn : 机器学习+指标评价等,经典机器学习相关方法的熟练使用系统开发：
  - 数据库接入：
    - pymongo : mongodb操作
    - PyMySQL : mysql操作