



School of Economics and Management, Beihang University

Python数据分析

课程介绍与Python开发环境

- 为什么叫Python数据分析

- 学通Python面向对象编程
- 掌握常见数据分析方法

- 面向对象编程

- 面向对象
- 设计模式
- 并行并发
- 网络编程

- 数据分析技术

- 科学计算
- 统计建模
- 可视化
- Tabular数据
- 自然语言理解
- 图像处理
- 声音处理



- 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

- 有了一定的专业视角
 - 数理基础更加扎实
 - 问题规模和难度相应提高
 - 作业的自由度不断增加

- 课程的目的

- 熟练掌握Python面向对象编程
 - 熟练使用Python完成多类数据分析
 - 自行阅读并学习**文档**
 - **重数据分析的内容与作业设置**
 - **重应用的编程训练**

```
toJSON: Function(object) {  
  var type = typeof object;  
  switch(type) {  
    case 'undefined':  
    case 'Function':  
    case 'unknown': return;  
    case 'boolean': return object.toString();  
  }  
  if (object === null) return 'null';  
  if (object.toJSON) return object.toJSON();  
  if (object.ownerDocument === document) return;  
  var results = [];  
  for (var property in object) {  
    var value = Object.toJSON(object[property]);  
    if (value !== undefined)  
      results.push(property.toJSON() + ':' + value);  
  }  
  return '[' + results.join(',') + ']';  
},  
  
keys: Function(object) {  
  var keys = [];  
  for (var property in object)  
    keys.push(property);  
  return keys;  
},
```

-
- A circular collage of various electronic and communication icons. The central area is filled with smaller icons like a globe, a film strip, a person at a computer, and a house. Surrounding this center are five larger, rounded square icons: a green one with a speaker, a blue one with a camera, a yellow one with headphones, a grey one with a laptop, and an orange one with a television. The entire composition is set against a light grey background with a subtle pink circular border.

• 主要内容

- Python编程基础
- 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
- 异步IO
- 科学计算
- 统计建模
- 网络编程
- **数据库**

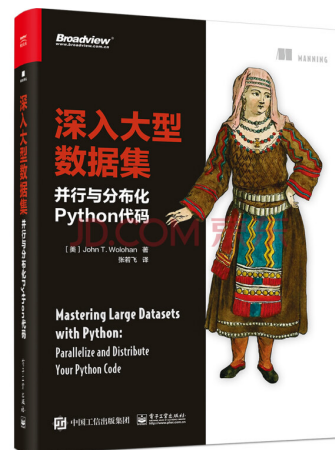
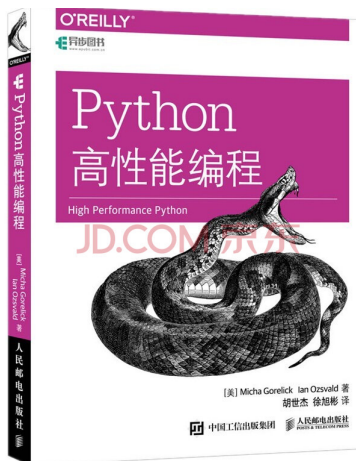
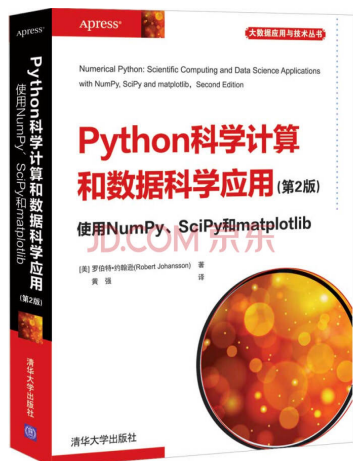
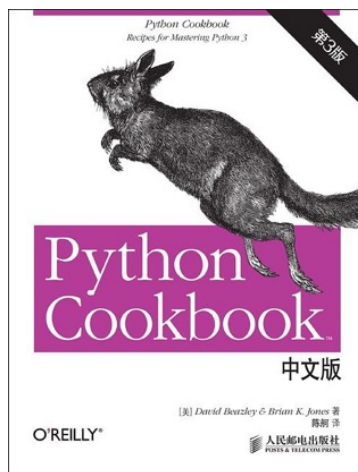


多模态数据分析任务
演示+作业

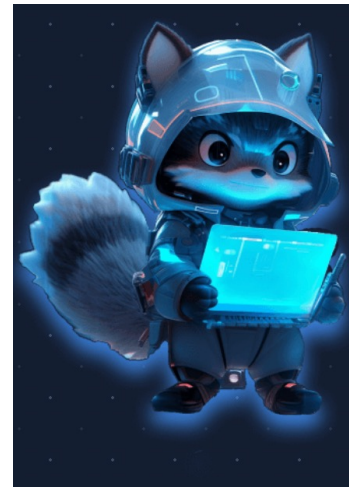
- 不建议购买相关书籍
 - 一般厚而贵
 - 相关内容极易失效（技术发展过快）
 - 编程库等的说明文档和实现示例网上更新最快
- 多上网查资料
 - 程序员们会分享很多心得
 - [csdn](#), [stackoverflow](#)
- 实际问题驱动
 - 学技术而非上“课”
 - 多动手
 - 将有力地支撑所属专业



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



- 语言建模取得了显著进展
 - 代码理解
 - 文本到代码生成
- 一些典型的工具
 - copilot
 - <https://github.com/features/copilot>
 - **codeium**
 - <https://codeium.com>
 - **代码小浣熊**
 - <https://raccoon.sensetime.com/code>
 - 一般有Vscode插件
 - **文心一言**等通用大模型
 - HuggingChat
 - <https://huggingface.co/chat>



- 考核

- **课程考勤** + 平时作业 + 期末考试

- 平时作业

- 一般每周一次
 - 结合当周内容完成某个数据分析相关的任务
 - **通过附加题等设定难度等级**

- 课下自学

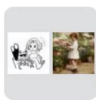
- 可能引入一些Python的第三方包或工具
 - **需要通过文档、大模型以及搜索引擎来自学其使用**
 - 培养主动解决编程问题的能力

- 期末考试

- 闭卷
 - 主要考查面向对象相关知识点



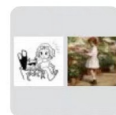
- 李瑾茜
- 陈飞铭
- 大家的“老”朋友（记得朋友圈“屏蔽”她们）
- 均有丰富的Python多模态数据分析经验
- 负责部分作业设计与作业检查（将100%覆盖）



群聊: python数据分析 2024-上午12节



该二维码7天内(3月6日前)有效, 重新进入将更新



群聊: python数据分析
2024-下午67节



该二维码7天内(3月6日前)有效, 重新进入将更新

- 答疑方式

- **大模型**

- 课前课后

- 一般会早到15分钟左右

- 办公室：新主楼A1006

- 周二下午、周三下午、周四上午

- 邮件：jichang@buaa.edu.cn

- 助教

- 通过课程群答疑
 - 建议周末

- 面向过程

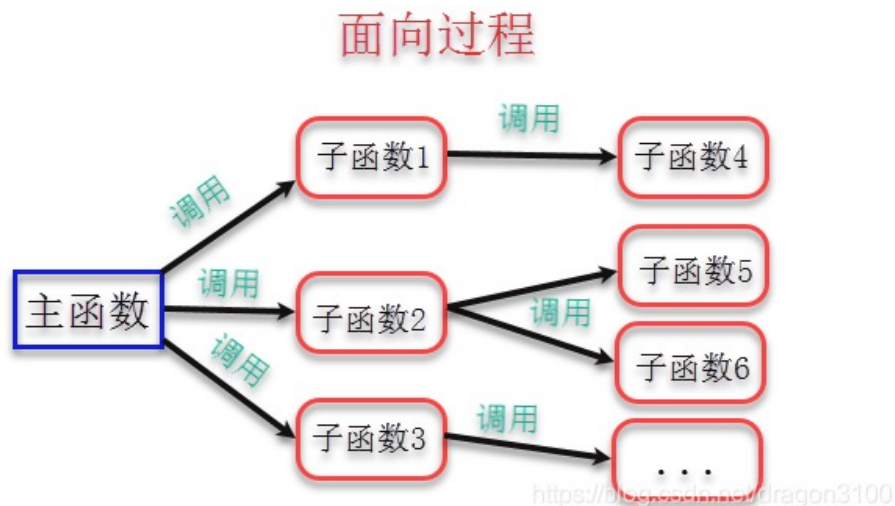
- 解决某个问题的流程
- 1+1

- 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

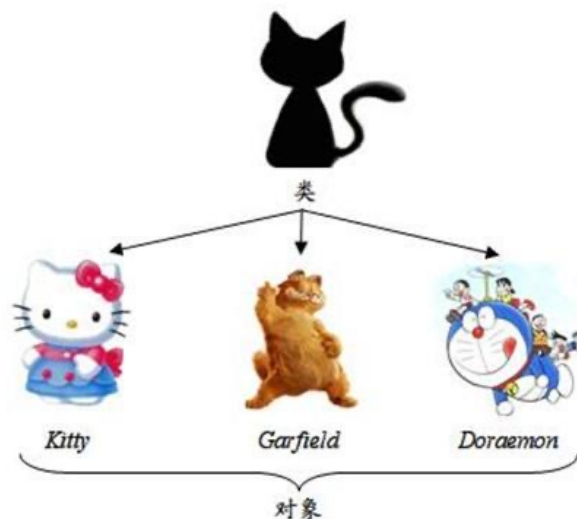
• 基本特点

– **封装**

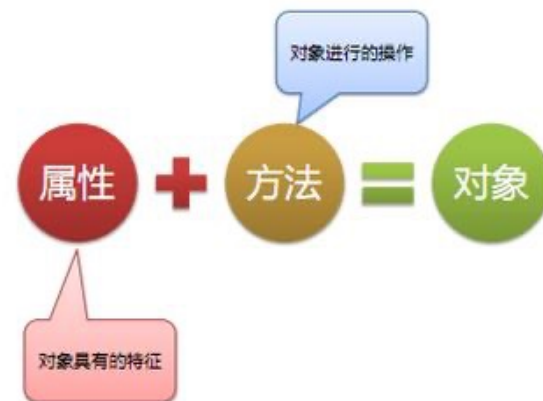
– **继承**

– **多态**

• Shape示例



对象的组成部分



• 数据

—数字、文本、图像、声音、视频

• 数据分析

—从数据中发现规律，挖掘知识，得到洞见

—数据采集

—数据预处理

—描述性分析

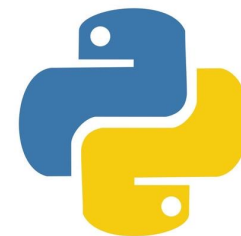
—建模分析

—预测应用

—决策支撑























- 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 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).



为什么以Python为例

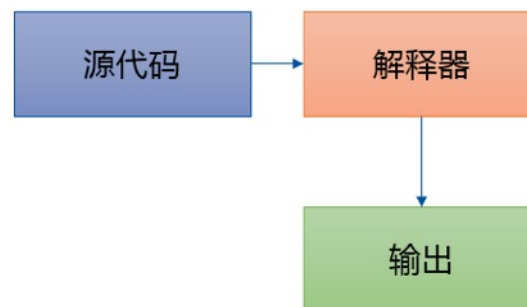
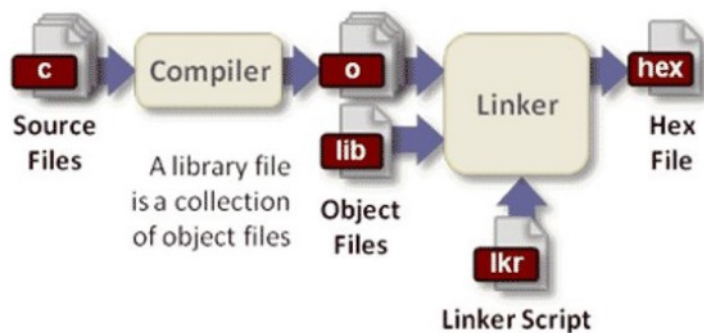


- 数据科学的重要性
- 人工智能的风靡

Feb 2024	Feb 2023	Change	Programming Language		Ratings	Change
1	1			Python	15.16%	-0.32%
2	2			C	10.97%	-4.41%
3	3			C++	10.53%	-3.40%
4	4			Java	8.88%	-4.33%
5	5			C#	7.53%	+1.15%
6	7	^		JavaScript	3.17%	+0.64%
7	8	^		SQL	1.82%	-0.30%
8	11	^		Go	1.73%	+0.61%
9	6	v		Visual Basic	1.52%	-2.62%
10	10			PHP	1.51%	+0.21%
11	24	^		Fortran	1.40%	+0.82%
12	14	^		Delphi/Object Pascal	1.40%	+0.45%
13	13			MATLAB	1.26%	+0.27%
14	9	v		Assembly language	1.19%	-0.19%
15	18	^		Scratch	1.18%	+0.42%
16	15	v		Swift	1.16%	+0.23%
17	33	^		Kotlin	1.07%	+0.76%
18	20	^		Rust	1.05%	+0.35%
19	30	^		COBOL	1.01%	+0.60%
20	16	v		Ruby	0.99%	+0.17%

• 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

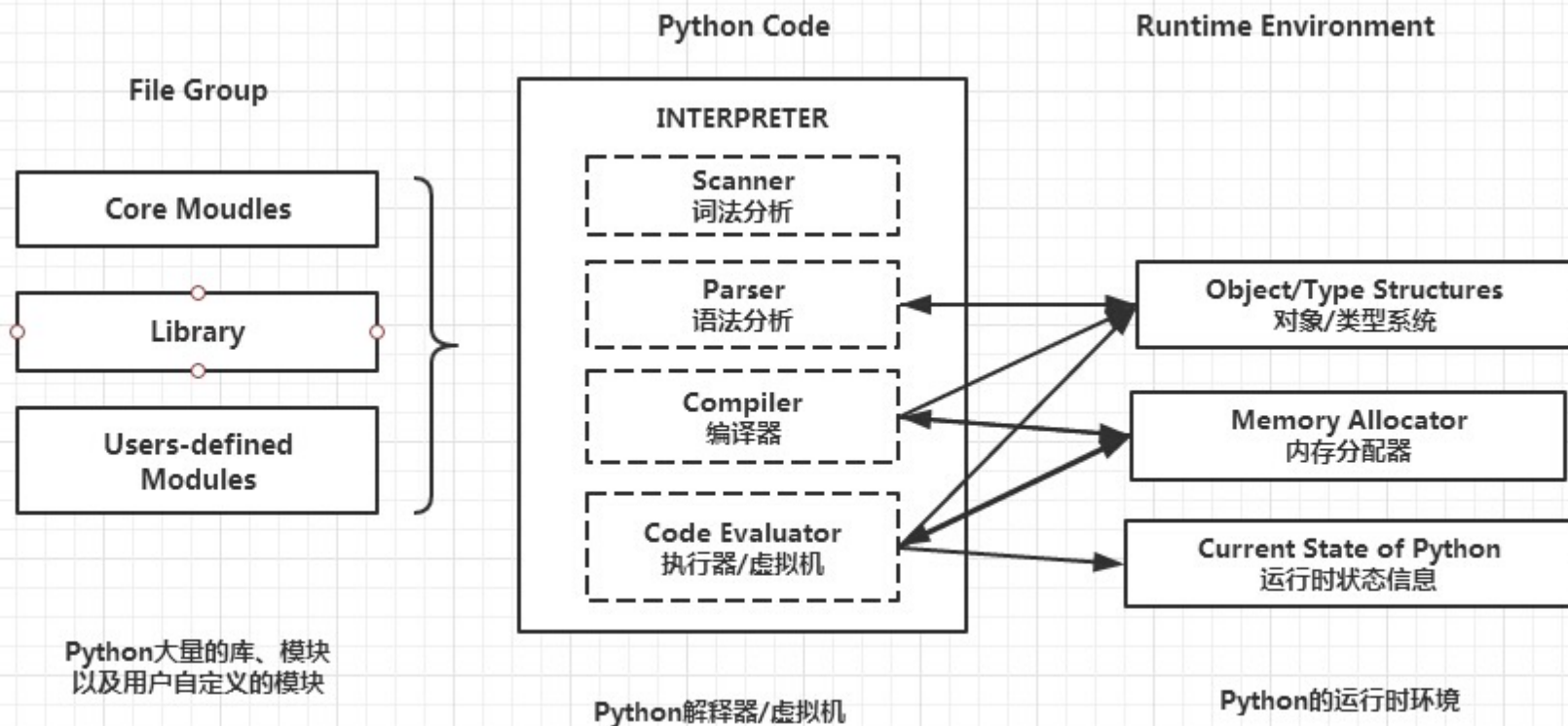
编译方式：运行速度快，但任何一个小改动都需要整体重新编译。可脱离编译环境运行。代表语言是C语言。



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。

Python解释器



- **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

- 语言版本

- Python 3 (上课演示的例子一般是3.11，最新的版本是3.12)
- 下载并安装对应平台的解释器

- 集成开发环境vscode

- 智能提示，使用大量核心库或第三库时较为方便
- 代码格式智能整理
- 大模型插件
- 其他插件



- Python包管理工具

- pip(pip3)

- pip install package

- # 最新版本

- **pip install package==1.0.4**

- # 指定版本

- pip install 'package>=1.0.4'

- # 最小版本

- pip uninstall package

- pip show package

- 从github安装

- pip install git+https://github.com/openai/whisper.git

- Anaconda
 - 需要安装(<https://www.anaconda.com>)
 - 虚拟环境管理非常方便
 - conda install package
 - conda remove package
 - conda env list #列出当前所有conda环境
 - **conda create --name 环境名称 #创建一个新的conda环境**
 - **conda activate #激活一个conda环境**
 - **conda deactivate #退出当前激活的conda环境**
 - conda env remove --name 环境名称 #删除conda环境

• Jupyter Notebook Interface

- It offers fast, **interactive new ways** to prototype and explain your code, explore and **visualize your data**, and share your ideas with others.
- 安装: **pip install notebook**
- 启动: **jupyter notebook**



- 1. 安装Python3。
- 2. 尝试用pip安装matplotlib等包，或者将其卸载。
- 3. 安装Anaconda，并利用conda创建一个虚拟环境pweek1，并在该环境中利用conda安装包jieba。
- 4. 安装Jupyter Notebook Interface，并尝试启动。
- 5. 利用某个大模型，生成一段绘制高斯分布曲线的代码，并在Jupyter Notebook中执行，观察代码是否能够运行，以及可视化的结果是否正确；如果能够调整均值和方差，观察绘制结果是否发生变化。
- 6.（附加）创建github等代码托管账户，管理和备份每周作业。
- 7.（附加）了解并关注SI里的常用包（本门课应该会全部涉及到）。

• 本课程后续用到的包

– 基础功能或技巧：

- tqdm：进度条，训练或者数据加载非常有用
- json（标准库）/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（标准库）：结果序列化存储
- argparse：交互参数解析

– 数据分析：

- numpy
- scipy
- pandas

– 可视化：

- matplotlib：最常用的绘图工具
- seaborn：辅助matplotlib使用
- 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)

- 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：
 - fastHan/jieba：分词工具
 - gensim:话题模型及word2vec嵌入等
 - scapy: 自然语言处理，实体识别等
 - pyltp: 中文处理，分词，实体，词性等
- 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

- 金融及量化交易
 - pyfolio : 交易策略等
 - finrl : 强化学习在股价预测与交易中的应用
- 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 :
 - openai : 大模型api会使用该库