

课程绪论与概述

张建章

阿里巴巴商学院

杭州师范大学

2024-09



1 课程考核说明

2 关于课程

3 计算思维

4 计算机与计算机语言

5 Python 编程环境配置

6 Python 初学者指南

7 Python 编程实例

根据教学大纲要求，本课程的考核办法为：

$$\begin{aligned}\text{总成绩} = & \text{期末成绩} \times 50\% + \text{日常作业} \times 30\% \\ & + \text{日常考勤} \times 10\% + \text{课堂表现} \times 10\%\end{aligned}$$

其中，期末考试采用上机考试形式。

课程名称：《Python 程序设计》

课程目标：

- ① 掌握 Python 编程语言；
- ② 培养“计算思维”；
- ③ 通过程序设计高效解决实际问题。

授课方式：上机实验为主，主要基于 Jupyter-lab 交互式编程教学

作业提交：坚果云在线提交 Jupyter Notebook 文件（后缀为.ipynb）

商科同学为什么要学 **Python**——赋能数字经济研究与实践：

2. 关于课程



数据分析



舆情分析



可视化



量化投资

什么是计算思维

计算思维 (computational thinking): 计算思维 (Computational Thinking, CT) 是一种通过运用计算机科学的基本概念来解决问题、设计系统和理解人类行为的过程。计算思维的核心包括以下四个方面：

- 分解 (Decomposition): 将复杂问题分解为更小、更易于处理的部分，从而使问题更加可理解和可管理；
- 模式识别 (Pattern Recognition): 识别问题中的模式和规律，找出相似性，从而简化解决方案的开发过程；
- 抽象 (Abstraction): 提取问题的关键信息，忽略不必要的细节，以简化问题的表示和解决过程；
- 算法思维 (Algorithmic Thinking): 设计一个明确的、有步骤的解决方案 (算法)，使其可以被计算机执行，或为解决问题提供一种系统化的方法。

计算思维实例

案例: 规划一次度假

假设你正在计划一次家庭度假,这个过程可以应用计算思维的四个核心方面:

- 1. 分解 (Decomposition):**首先,将度假计划分解为更小的任务。例如,你需要选择目的地、确定预算、预订住宿和安排交通。每一个任务都是一个独立的子问题,可以分别解决。
- 2. 模式识别 (Pattern Recognition):**在分解任务后,下一步是识别模式。例如,你可能注意到,每次度假都涉及类似的步骤,如选择住宿和交通工具。这些模式帮助你更有效地进行计划,因为你可利用以前的经验来简化当前的任务。
- 3. 抽象 (Abstraction):**在规划过程中,你需要忽略不相关的细节,专注于关键因素。例如,在选择目的地时,你可能不需要考虑所有可能的天气条件,而只需要关注主要的气候类型和旅游季节。这种简化使得问题更加可控。
- 4. 算法思维 (Algorithmic Thinking):**最后,根据分解、模式识别和抽象的结果,制定一个明确的步骤来完成度假计划。这包括决定什么时候出发、如何预订机票和酒店,以及每天的活动安排。这些步骤应当详细到足以让任何人根据这些步骤顺利完成计划。

总结

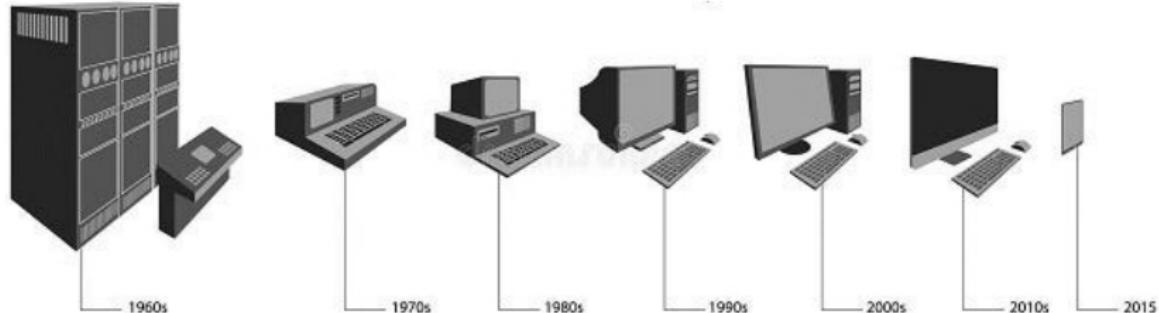
通过应用计算思维的四个核心方面,度假规划变得系统化和高效。首先,通过分解,将复杂问题拆分为可管理的部分;然后,通过模式识别,利用已知的解决方法加速当前任务;通过抽象,聚焦于重要信息而忽略无关细节;最后,通过算法思维,制定一个可执行的计划。

培养计算思维的重要性

- **提升问题解决能力：**在商业环境中，经常需要应对复杂的业务问题，如市场分析、财务建模和供应链管理等。通过培养计算思维，可以学会如何将复杂问题分解成更小的部分，找到问题中的模式，并设计出有效的解决方案；
- **数据分析和决策制定：**在大数据时代，商业决策越来越依赖于对大量数据的分析和解读。计算思维能够提升数据处理能力，使得在数据中识别模式、预测市场趋势或评估财务风险时更加高效和准确；
- **创新与自动化：**计算思维不仅帮助解决现有问题，还能激发创新。通过理解和应用算法设计，可以开发新的工具和方法来自动化重复性的业务流程，提高工作效率和创造力；
- **跨学科应用：**计算思维的核心方法，如分解、抽象和模式识别，能够在不同的学科和领域之间迁移和应用；

计算机

计算机是能够被编程以自动执行一系列算术或逻辑操作的机器，即，能够按照事先存储的程序（可编程性），自动、高速地对数据进行输入、处理、输出和存储的系统（功能性）。



- 早期计算设备：查尔斯·巴贝奇在 1820 年代设计的差分机；
- 电子计算机：1940 年代，电子数值积分计算机（ENIAC）问世；
- 个人计算机和互联网：80 年代初，个人计算机出现，21 世纪初，互联网普及；
- 现代计算机：向更高的速度、更大的存储容量和更高的集成度发展，如云计算和量子计算。

计算机语言

计算机语言是一种用于向计算机传达指令的符号系统。通过这些语言，人们可以编写计算机程序，这些程序告诉计算机如何执行特定的任务。

- **Python:** 一种通用的编程语言，广泛用于数据分析、机器学习、人工智能、web 开发等；
- **JavaScript:** 一种主要用于前端开发的语言，常用于构建动态和交互式网页；
- **Java:** 是一种面向对象的编程语言，广泛用于企业级应用、安卓移动应用和大型系统开发；
- **SQL:** 结构化查询语言，用于管理和操作关系数据库，是处理大规模数据集和进行数据分析的重要工具。

常用编程语言



图 1: 常用编程语言

计算机语言的特点

- 精确性和确定性：计算机语言是高度精确的，用于向计算机传达明确的指令；
- 语法和语义：计算机语言的语法和语义固定不变，专门用于消除歧义并确保计算机能准确执行任务；
- 表达能力有限：主要用于执行明确的计算任务和操作，缺乏对情感和复杂人类思想的表达能力；
- 结构和抽象：计算机语言通常使用固定的词汇和严格的语法规则来定义操作，让程序员能够更好地控制计算机硬件。

静态语言与脚本语言

静态语言通常使用编译器并进行类型检查以确保代码的安全性和性能。而脚本语言往往使用解释器，具有动态类型，可以更方便地进行快速开发和迭代。

静态语言（Static Languages）：静态语言通常是指在编译时需要明确类型的编程语言。变量的类型在编写代码时就已经确定，并且在整个程序的生命周期中不会改变。这种类型的语言在编译阶段会进行类型检查，这有助于在代码运行之前捕获潜在的错误。例如，C、C++ 和 Java 都是典型的静态语言。

脚本语言（Scripting Languages）：脚本语言是一种解释执行的编程语言，通常用于自动化任务、处理数据或在 web 开发中生成动态内容。与静态语言不同，脚本语言通常具有动态类型，这意味着变量的类型是在运行时确定的。脚本语言一般不需要预编译为机器码，而是通过解释器逐行执行。例如，Python、JavaScript 和 PHP 都是常见的脚本语言。

编译与解释

编译和解释是两种不同的程序执行方式。编译器在执行之前将整个源代码一次性转换为机器码，而解释器逐行执行代码。编译语言通常具有更好的性能，而解释语言则更具灵活性和开发速度。

编译（Compilation）：编译是将高级编程语言代码转换为机器码的过程。编译器会在程序运行之前将整个源代码翻译为机器可执行的文件。

解释（Interpretation）：解释是逐行读取和执行源代码的过程。解释器在代码运行时实时翻译和执行代码，这使得开发者能够快速测试和修改代码，因为不需要在每次修改后重新编译整个程序。



编译



解释

编程的基本原则

编写程序的主要原则是确保代码简洁、可读、易于维护，并能有效地执行任务。以下是一些核心编程原则，这些原则有助于开发人员编写高质量的代码，减少错误，提高代码的可读性和可维护性：

- KISS (Keep It Simple, Stupid!);
- DRY (Don't Repeat Yourself);
- 单一职责原则 (Single Responsibility Principle): 每个模块或函数应只负责一个特定功能，这样可以减少耦合，提高代码的可维护性；
- 文档化 (Document Your Code): 编写清晰的代码注释，帮助其他开发者理解代码的意图和功能，尤其在团队合作中尤为重要。

Python 编程语言简介

Python 是一种高级、通用的编程语言，由 Guido van Rossum 于 1991 年首次发布。它以简洁、易读的语法和广泛的应用领域而闻名。

- 易于学习和使用；
- 广泛的应用领域：数据科学、人工智能、机器学习、Web 开发、自动化脚本、数据分析等；
- 丰富的库和社区支持：Python 拥有庞大的标准库和第三方库，这些库为图形处理、数据分析、机器学习、Web 开发等提供了强大的支持；
- 跨平台兼容性；
- 动态类型和自动内存管理：变量类型在运行时确定使 Python 更灵活，能更快地开发和测试代码。Python 还具有自动内存管理功能，开发者不需要手动管理内存分配，这减少了编程中的常见错误。

Python 是一种非常适合商科学生学习的编程语言，通过学习 Python，商科学生不仅能够提升数据分析能力和自动化水平，还能获得在现代商业环境中所需的技术技能，为未来的职业发展打下坚实的基础。



图 2: 2024 年度 IEEE Spectrum 编程语言排行榜

本地编程环境

Anaconda 是一个面向科学计算、数据科学和机器学习的 Python 发行版。它不仅包括 Python，还集成了用于数据分析、机器学习和科学计算的超过 1500 个开源软件包。Anaconda 的主要工具包括 Conda、Jupyter Notebook、Spyder IDE 等。使用 Anaconda 便于更专注于学习 Python 编程和数据分析技能，而不必担心环境配置和依赖管理的问题。

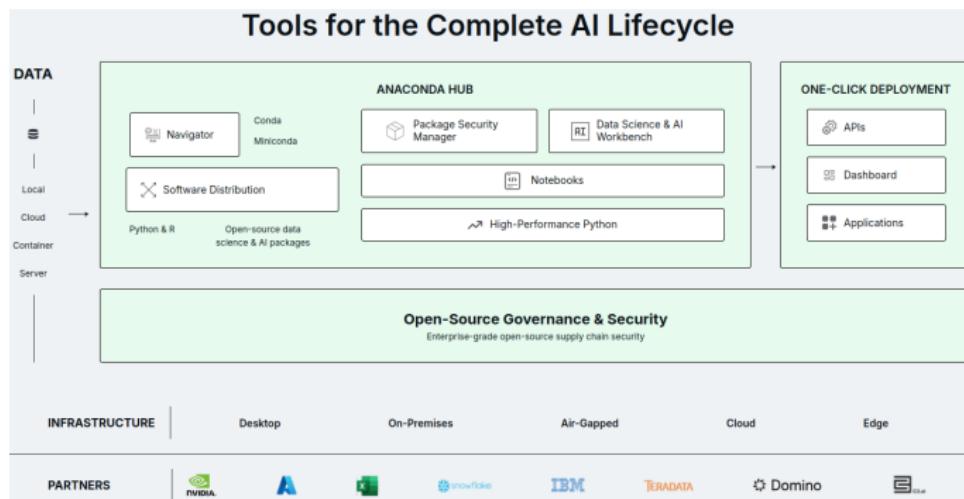


图 3: Anaconda 面向 AI 生命周期的工具集

Windows 安装 Anaconda

判断 32 位和 64 位 Windows



在线视频

Jupyter-lab 基本用法



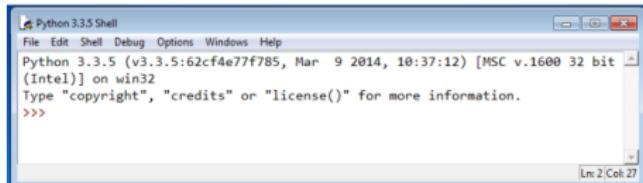
[在线视频, 文字说明 \(For Windows\)](#); [在线视频, 文字说明 \(For Mac\)](#)

在线编程环境

在线 Python 编程环境（如魔搭社区、Kaggle、Google Colab 等）提供了多种独特的优势，尤其是对于初学者和需要灵活工作环境的用户。这些环境可以通过标准的网络浏览器直接访问，无需在本地安装 Python 或相关工具，极大地降低了设置的复杂性。一些平台还支持高级计算资源（如 GPU 和 TPU），非常适合需要大量计算能力的任务，如机器学习模型的训练。



让 Python 编程更高效的工具



A screenshot of the Python 3.3.5 Shell window. The menu bar includes File, Edit, Shell, Debug, Options, Windows, Help. The status bar shows "Python 3.3.5 (v3.3.5:62cf4e77f785, Mar 9 2014, 10:37:12) [MSC v.1600 32 bit (Intel)] on win32". The command line shows "Type "copyright", "credits" or "license()" for more information." and a prompt "=>". The bottom status bar indicates "Ln 2 Col 27".

原生 Python 和 IDE



能跑的车架子



集成开发环境



跑车

为了学 Python 我需要什么样的电脑



没有必要购买高端电脑，如外星人



市面上普通的电脑即可用于本课程学习

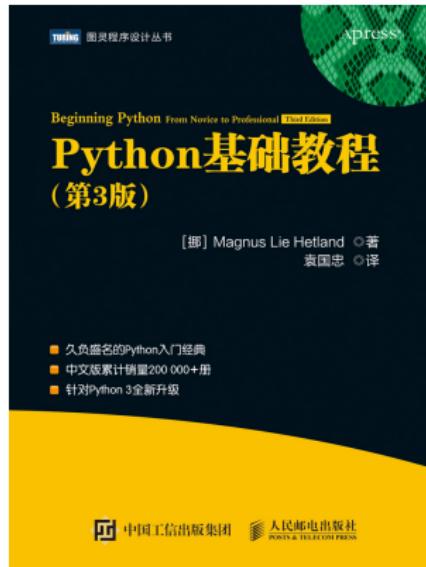
学习资源



《程序设计基础》课程网站

课程讨论区 

课堂和课后互动



参照课本勤奋练习



善于搜索互联网

善用生成式人工智能工具

生成式人工智能（Generative AI）工具，如 ChatGPT、Claude、Copilot、Cursor，能够极大地提升 Python 初学者的学习效率。这些工具可以自动生成代码片段，帮助初学者快速理解如何实现特定功能，并学习 Python 的语法和最佳编码实践。我国近年来也涌现出一批出色的生成式人工智能工具，如，阿里巴巴的通义千问、月之暗面的 Kimi、字节跳动的豆包、百度的文心一言等。



计算机编程的基本方法

- ① 输入 (Input): 终端命令行交互, 文件, 网络;
- ② 处理 (Processing): 对输入数据进行计算并产生输出结果的过程;
- ③ 输出 (Output): 通过终端命令行, 文件, 网络等输出结果。



计算实例：体质指数

体质指数 (Body Mass Index, BMI) = $\frac{\text{体重 (kg)}}{\text{身高 (m)}^2}$ 。

BMI 值是一个中立而可靠的指标，是国际上常用的衡量人体胖瘦程度以及是否健康的一个标准。

计算机：

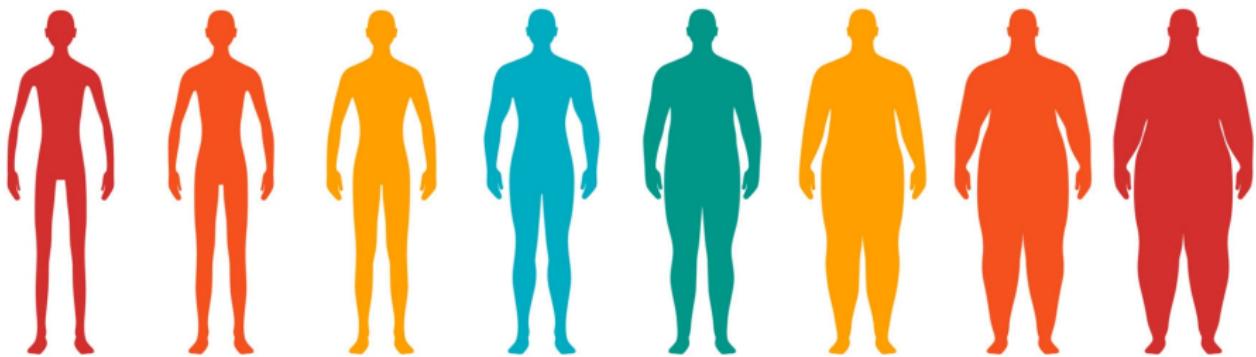
- ① 接收用户输入的身高、体重数据；
- ② 根据公式计算；
- ③ 输出体质指数。

用户：

- ① 输入身高、体重数据；
- ② 查看计算结果，对照量表；

BMI 对照表

BODY MASS INDEX (kg/m²)



| | | | | | | | |
|-----------------|-------------------|---------------|-----------|------------|---------------|----------------|-----------------|
| < 16 | 16 - 17 | 17 - 18.5 | 18.5 - 25 | 25 - 30 | 30 - 35 | 35 - 40 | > 40 |
| Severe Thinness | Moderate Thinness | Mild Thinness | Normal | Overweight | Obese Class I | Obese Class II | Obese Class III |

用 Python 计算 BMI

```
Height = float(input("请输入身高 (m): "))
Weight = float(input("请输入体重 (kg): "))

BMI = round(Weight/Height**2, 2)

if BMI>=25:
    print("BMI 指数为", BMI, " 体质偏重")
elif BMI<=18.5:
    print("BMI 指数为", BMI, " 体质偏轻")
else:
    print("BMI 指数为", BMI, " 正常")
```

THE END