算法入门

TEACHING COURSEWARE POWPOINT

授课时间：2025.07.25

目录

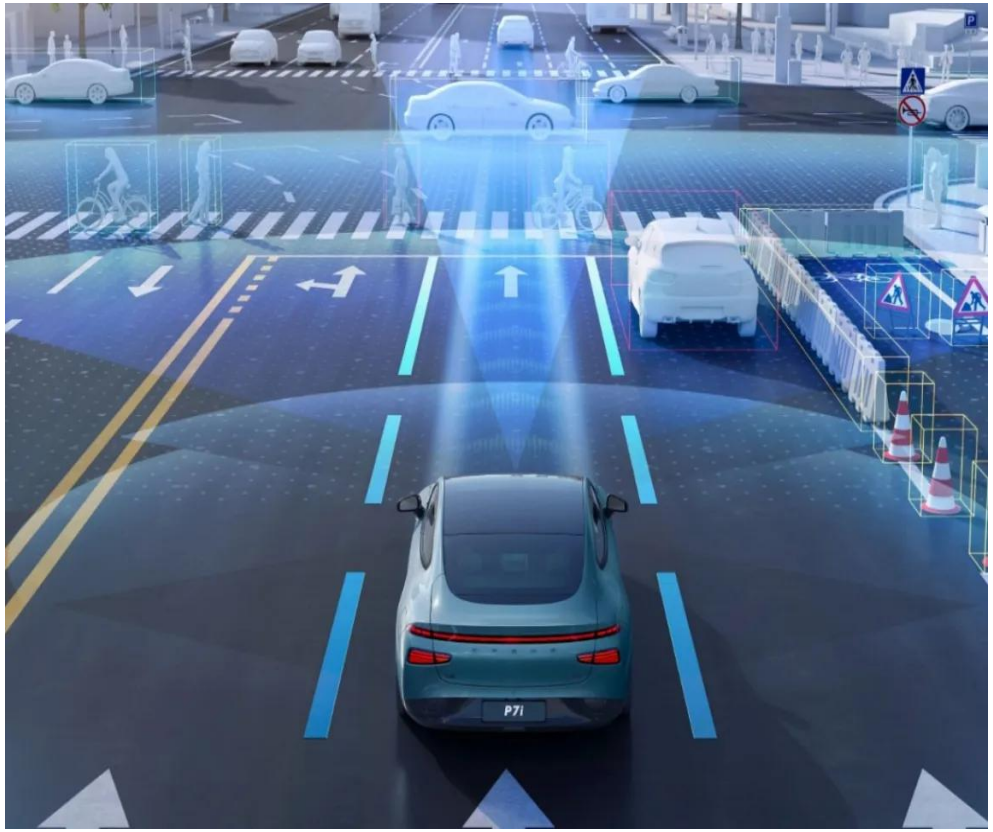
01

什么叫算法

TEACHING
COURSEWARE

TEACH

什么叫算法



智驾

煎鸡蛋

02

概念引入

TEACHING
COURSEWARE

TEACH

概念引入

算法（Algorithm）是指解题方案的准确而完整的描述，是一系列解决问题的清晰指令，算法代表着用系统的方法描述解决问题的策略机制。也就是说，能够对一定规范的输入，在有限时间内获得所要求的输出。如果一个算法有缺陷，或不适合于某个问题，执行这个算法将不会解决这个问题。不同的算法可能用不同的时间，空间或效率来完成同样的任务。一个算法的优劣可以用空间复杂度与时间复杂度来衡量。

算法中的指令描述的是一个计算，当其运行时，能从一个初始状态和（可能为空的）初始输入开始，经过一系列有限而清晰定义的状态，最终产生输出并停止于一个终态。

简言之，算法的核心是 “用确定、有限的步骤，规范地从输入到输出解决问题，且效率可量化”。这三个维度缺一不可：缺了确定性则指令无效，缺了有限性则无法落地，缺了目标与效率则失去存在意义。

03

三大表示

TEACHING
COURSEWARE

TEACH

自然语言描述

概念：用人类日常交流的语言（如中文、英文），按逻辑顺序描述算法的步骤、条件和结果。

特点：通俗易懂，无需专业知识即可理解，但需注意描述的精确性（避免歧义）；适合表达简单流程或初步梳理逻辑。

核心要素：包含“输入”、“步骤”、“条件判断”、“输出”。

自然语言描述

算法名称：煎鸡蛋
输入：生鸡蛋 1 个、食用油、锅、炉灶、铲子、盐（可选）
输出：煎熟的鸡蛋
步骤
准备阶段：将锅洗净，置于炉灶上；打开炉灶，中火预热锅 10–15 秒。
倒油：向锅中倒入少量食用油（覆盖锅底即可），等待 3–5 秒让油微热。
打入鸡蛋：将生鸡蛋磕入锅中，保持中火，让鸡蛋自然摊开。
煎制正面：观察鸡蛋，待蛋白完全凝固、边缘微焦（约 30 秒 – 1 分钟），判断是否需要翻面。
翻面：用铲子轻轻将鸡蛋翻面。
煎制反面：继续中火煎制 20–30 秒，至蛋白凝固、蛋黄达到预期熟度。
调味：在鸡蛋表面均匀撒上少许盐。
出锅：用铲子将鸡蛋盛入盘中，关闭炉灶，算法结束。

流程图

概念：用标准化的图形符号（如矩形、菱形、箭头等），可视化展示算法的执行流程、分支（条件判断）、循环等结构。

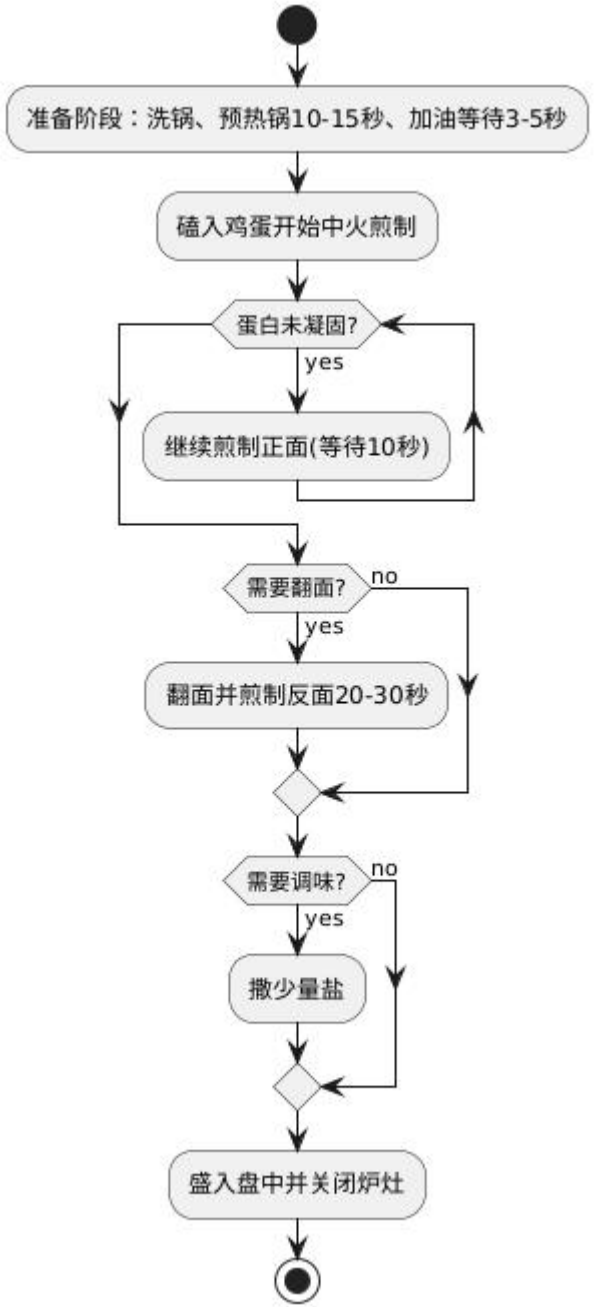
特点：逻辑直观，能清晰体现步骤的先后顺序、“如果… 就…”的分支关系、重复操作；适合展示复杂流程的结构。

核心要素：

基础符号：开始 / 结束框（椭圆形）、处理框（矩形，如 “倒油”）、判断框（菱形，如 “蛋白是否凝固”）、流程线（箭头，指示步骤顺序）。

逻辑关系：顺序执行、分支（判断后的不同路径）、循环（重复某步骤直到条件满足）。

流程图



形状	含义	示例（煎鸡蛋流程）
黑色实心圆（顶部）	起始节点	流程开始
圆角矩形（长条）	动作状态	洗锅、煎制、盛盘等具体操作
菱形框	判断节点（条件分支）	“蛋白未凝固？”“需要翻面？”的判断
箭头线	控制流（步骤执行顺序）	连接各节点，指示流程走向
黑色实心圆+外框（底部）	结束节点	流程结束

伪代码

概念：介于自然语言和编程语言之间的结构化描述语言，保留程序的核心逻辑（如条件、循环、函数），但不依赖具体编程语言的语法。

特点：兼顾可读性（像自然语言一样好懂）和逻辑性（像代码一样有结构）；
适合需要严谨描述算法的场景，方便转化为实际代码。
核心要素：
结构化语句：if-else（条件判断，如 “如果需要翻面，则执行翻面操作”）、while（循环，如 “直到蛋白凝固”）、函数（封装流程，如 “煎鸡蛋 ()”函数）。
简洁性：省略细节

伪代码

```
煎鸡蛋():
    // 准备阶段
    锅 = 洗净(锅)
    放置(锅, 炉灶)
    打开(炉灶, 中火)
    等待(10-15秒)  // 预热锅

    // 倒油
    倒入(锅, 食用油, 少量)
    等待(3-5秒)  // 油微热

    // 打入鸡蛋并煎制
    打入(锅, 生鸡蛋)
    状态 = "正面煎制中"

    // 循环判断正面是否煎好
    当 状态 == "正面煎制中" 时:
        若 鸡蛋.蛋白凝固() 且 鸡蛋.边缘微焦():
            状态 = "判断是否翻面"
        否则:
            等待(10秒)  // 继续煎制

    // 处理翻面
        用铲子翻面(鸡蛋)
        等待(20-30秒)  // 煎制反面

    // 调味与出锅
    若 需要调味:
        撒入(锅, 盐, 少量)
    盛入(鸡蛋, 盘子)
    关闭(炉灶)
    返回 煎好的鸡蛋

// 执行算法
```

04

复杂度分析

TEACHING
COURSEWARE

TEACH

复杂度

1. 时间复杂度：

时间复杂度是衡量算法执行效率的指标，描述算法执行所需的基本操作次数（如判断、计算、赋值等）随输入规模 n（即问题中数据量的大小，如数组长度、数据个数等）增长的趋势。

核心特征：关注 “输入规模变大时，操作次数如何增长”，而非 “具体执行时间”，通常以 “最坏情况下的增长趋势” 为衡量标准。

2. 空间复杂度：

空间复杂度是衡量算法资源消耗的指标，描述算法在执行过程中所需的存储空间（如变量、数组、临时数据等占用的内存）随输入规模 n 增长的趋势。核心特征：关注 “输入规模变大时，所需空间如何增长”，同样忽略具体数值，仅关注增长趋势。

3. 大 O 符号：

大 O 符号是描述复杂度（时间或空间）增长趋势的数学工具，其标准定义为：

设算法的操作次数（或存储空间）为 T(n)（n为输入规模），若存在两个常数 c > 0（正数）和 n_0（某个足够大的输入规模），使得当 n >= n_0 时，始终满足 T(n)<= c*f(n)，则称该算法的复杂度为O(f(n))

核心含义：O（f(n)）表示算法复杂度的 “增长趋势上限”—— 即当输入规模足够大时，操作次数（或空间）的增长速度不会超过 f(n) 的增长速度。

时空复杂度

输入规模，n 就是 “问题的大小”
时间复杂度：“步骤数随任务量的变化”比如做任务时，“需要多少步” 会随着 “任务变难”（n 变大）怎么变？
例子 1：O(1)（常数时间）
任务：从班级名单里找出学号为i的同学的名字。
不管班级有 30 人(n=30）还是 100 人（n=100），步骤都是 “直接看第 i 个”，只需要 1 步。
规律：步骤数不随 n 变化，固定为常数 —— 这就是 O(1)。（加减乘除）
例子 2：O(n)（线性时间）
任务：从班级里找出最高的人。
最坏情况：必须把每个人都看一遍（30 人看 30 次，100 人看 100 次）。
规律：步骤数随 n 成 “正比例增长”（n 翻倍，步骤数也翻倍）—— 这就是 O(n)。
例子 3：O(n^2)（平方时间）
任务：班级里每个人都要给其他人送一张贺卡（自己不送自己）。
30 人时：每个人送 29 张，总共约 30×30=900 张（近似）；100 人时：约 100×100=10000张。
规律：步骤数随 n 成 “平方增长”（n 翻倍，步骤数翻 4 倍）—— 这就是 O(n^2)。

时空复杂度

空间复杂度
例子：
0(1)（常数空间）
任务：计算班级所有人的平均身高。
只需要一个变量记录 “总身高” 和 “人数”，不管 30 人还是 100 人，需要的空间都不变 —— 这就是空间复杂度 0(1)。
例子：0(n)（线性空间）
任务：把班级所有人的名字抄到一张新名单上。
30 人需要 30 个位置，100 人需要 100 个位置，空间随 n 正比例增长 —— 这就是空间复杂度 0(n)。

总结

TEACHING
COURSEWARE

启航舱
中小学拔尖创新人才培养中心

总结

DAILY REPORT INTRODUCTION

算法是解决问题的准确完整描述，表现为一系列清晰的指令：能对规范输入在有限时间内产生预期输出，其核心在于“确定性（步骤明确）、有限性（步骤可数）、目标与效率可量化”，缺一不可。生活中煎鸡蛋的步骤、智驾系统的决策逻辑等，都是算法的具体体现。

自然语言描述：用日常语言按逻辑顺序描述步骤（如煎鸡蛋的“预热锅→倒油→煎制”等），通俗易懂但需避免歧义，核心包含输入、步骤、条件判断、输出。

总结

TEACHING
COURSEWARE

启航舱
中小学拔尖创新人才培养中心

总结

DAILY REPORT INTRODUCTION

流程图：用标准化符号（椭圆形表示开始/结束、矩形表示处理步骤、菱形表示条件判断、箭头指示流程）可视化展示逻辑，清晰体现顺序、分支、循环关系，适合复杂流程。

伪代码：介于自然语言与编程语言之间，保留结构化逻辑（如if-else条件、while循环），兼顾可读性与严谨性，便于转化为实际代码。

总结

TEACHING
COURSEWARE

总结

DAILY REPORT INTRODUCTION

时间复杂度：描述执行步骤随输入规模（n）的增长趋势，用大O符号表示（如O(1)常数时间、O(n)线性时间、O(n²)平方时间），反映算法“快慢”。

空间复杂度：描述临时存储空间随n的增长趋势（如O(1)常数空间、O(n)线性空间），反映“内存占用多少”。

两者共同作为评估核心，实际应用中常需权衡（如“用空间换时间”）。