Creative Education Studies 创新教育研究, 2026, 14(11), 188-197
Published Online November 2026 in Hans. https://www.hanspub.org/journal/ces
https://doi.org/10.12677/ces.2026.1411877

Hans汉斯

生成式人工智能支持下《Python程序设计》课程分层支架教学模式构建与实践

李癸雨，丁晓峰，杨菊英，杨　铸，薛丽莹

四川大学锦江学院计算机学院，四川　眉山

收稿日期：2026年9月21日；录用日期：2026年11月2日；发布日期：2026年11月17日

摘　要

针对生成式人工智能背景下学生编程基础差异显著、AI依赖加剧及知识迁移能力不足等问题，本文围绕《Python程序设计》课程教学，构建“分层支架－AI透明化－纠错共享－项目链进阶”四维联动教学模式。通过设置分层任务与差异化学习支架，满足学生个性化学习需求；引导学生规范使用生成式AI，强化代码解释、结果核验与过程记录；借助错误提交和调试复盘促进纠错经验共享；以递进式项目串联知识，提升学生的知识迁移与综合实践能力。同时，结合任务成果、学习过程、调试表现与项目评价，建立多元过程性评价机制。该模式有助于平衡生成式AI辅助与学生自主学习，为《Python程序设计》课程的差异化教学和人才培养提供参考。

关键词

生成式人工智能，Python程序设计，分层支架，递进式项目

Construction and Practice of a Tiered Scaffolding Instructional Model for the “Python Programming” Course Supported by Generative Artificial Intelligence

Guiyu Li, Xiaofeng Ding, Juying Yang, Zhu Yang, Liying Xue

School of Computer Science, Sichuan University Jinjiang College, Meishan Sichuan

Received: September 21, 2026; accepted: November 2, 2026; published: November 17, 2026

文章引用: 李癸雨, 丁晓峰, 杨菊英, 杨铸, 薛丽莹. 生成式人工智能支持下《Python 程序设计》课程分层支架教学模式构建与实践[J]. 创新教育研究, 2026, 14(11): 188-197. DOI: 10.12677/ces.2026.1411877

Abstract

In response to the significant differences in students' programming proficiency, increasing dependence on AI, and insufficient knowledge transfer abilities in the context of generative artificial intelligence, this study focuses on the teaching of the Python Programming course and constructs a four-dimensional instructional model integrating "tiered scaffolding, AI transparency, error-correction sharing, and progressive project-based learning." Tiered tasks and differentiated learning scaffolds are designed to meet students' individualized learning needs. Students are guided to use generative AI appropriately, with emphasis on code explanation, result verification, and process documentation. Error submission and debugging reflection are employed to promote the sharing of error-correction experiences. Progressive projects are used to connect and integrate knowledge, thereby enhancing students' knowledge transfer and comprehensive practical abilities. In addition, a diversified process-oriented assessment system is established by incorporating task outcomes, learning processes, debugging performance, and project evaluation. This model helps balance generative AI-assisted learning with students' independent learning and provides a reference for differentiated instruction and talent development in the Python Programming course.

Keywords

Generative Artificial Intelligence, Python Programming, Tiered Scaffolding, Progressive Projects

Copyright © 2026 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

随着生成式人工智能等技术的快速发展，Python 凭借语法简洁、可读性强、第三方库与开源生态丰富等优势，已成为人工智能和数据科学领域事实上的主流编程语言。在人工智能产业需求与专业人才培养方案的双重驱动下，高校对毕业生的 AI 技术能力提出了明确要求，越来越多高校将《Python 程序设计》纳入课程体系。Python 基础是否扎实，不仅影响学生考研深造后开展深度学习等科研工作，也影响其未来从事大数据研发工程师、AI 应用工程师等岗位的职业发展。当前，许多 AI 相关岗位要求从业者熟悉大模型基本原理，而 Transformer 等大模型架构、YOLO 等主流深度学习模型以及 PyTorch、TensorFlow 等框架，多以 Python 作为主要开发、训练与部署语言。因此，提升《Python 程序设计》课程教学质量，对学生科研能力与职业竞争力具有重要意义。

然而，大三学生编程基础差异显著：部分学生已接触 Java 等语言，因 Python 语法更简洁且同属面向对象体系，能较快掌握基础，零基础学生则相对滞后。同时，生成式 AI 的普及使学生倾向直接获取答案，报错后也常交由 AI 修改，缺乏自主思考与调试，易致错误累积。此外，课堂知识点多以孤立小题练习，学生难以在项目中迁移、串联与进阶。为此，本文构建"分层支架－AI 透明化－纠错共享－项目链进阶"四维联动教学模式，以平衡生成式 AI 使用与学生自主认知活动，规避 AI 依赖导致的认知替代与调试惰性，发挥其认知支架作用，为《Python 程序设计》课程的差异化教学与过程性评价提供设计参考。

2. 国内外研究现状

2.1. 生成式 AI 赋能编程教育：工具潜力与人机协同认知视角下的现实困境

教师、学生与智能机器在功能上互补，在过程中协同，共同促进教学系统的优化与演进。生成式 AI

为编程教育构建起“教师–学生–智能机器”三元交互的新型教学生态[1]，AI 可以快速生成代码片段、提供语法解释、即时反馈报错信息，降低编程入门的认知门槛，拓展个性化学习资源供给，为教师减负、为学生赋能。现有研究大多将生成式 AI 视作智能辅导支架，探讨其在代码生成、习题答疑、即时反馈层面的应用价值，证实 AI 能够减轻基础语法带来的认知负荷，帮助学生聚焦算法与逻辑设计等高阶任务。

但大量实证研究同时揭示显著风险：无约束地使用 AI 极易催生工具依赖，学生直接复制 AI 输出的代码，跳过自主分析、手动调试、逻辑推演环节，出现“能跑代码，但不懂原理”的浅层学习现象，削弱调试排错与问题拆解能力。从人机协同认知理论来看，AI 应当作为学习者认知的外延辅助，承担信息检索、样板生成等辅助性认知工作，而把需求分析、逻辑校验、方案取舍、故障排查留给学习者完成，形成“人主导、AI 辅助”的认知伙伴关系，而非 AI 替代人的思考过程。然而当前多数课程缺少对 AI 使用过程的教学设计：教师只能看到最终代码成果，无法观察学生如何提问、如何甄别、修改 AI 输出内容，人机交互过程处于“黑箱”状态，难以区分“有效利用 AI 拓展思路”和“直接复制答案完成任务”两类行为，无法开展针对性干预引导。部分研究提出要记录学生与 AI 交互痕迹、强化对 AI 输出的批判性审视，但尚未将过程透明化嵌入完整课程实践流程，缺少可落地的教学实施路径。

由此可见，仅仅放开 AI 工具使用或者简单禁止 AI，均无法达成育人目标，需要配套教学机制约束与引导人机交互行为，这正是本研究中 AI 透明化维度的理论出发点：要求完整留存提示词、AI 原始代码、人工修改版本、调试复盘记录，把黑箱式人机交互转化为可复盘、可评价的显性学习过程，保障学生在人机协同中的认知主体地位。

2.2. 支架式教学与分层教学研究

在教学实施中，理论理解与动手实践之间常存在转化断层。学生虽能理解概念与算法逻辑，却因编程经验不足和畏难情绪而不敢动手编码，致使学习滞留于书本层面和理论层面。然而，编程实践能力是学生未来专业发展与职业胜任的关键素养。基于此，教师应提供适切的学习支架[2]，引导学生依次经历“模仿–半模仿–独立”三个阶段；随着学生编程能力提升与动手畏难情绪缓解，逐步实施动态撤架，推动其由外部支持依赖转向自主编码与自我调节，最终实现独立编程能力的发展[3]。这一过程契合支架式教学与最近发展区理念，有助于实现从“扶”到“放”的教学过渡。

在班级教学情境中，学生知识基础、学习能力与学习需求存在显著异质性。在资源与时间有限的约束下，教师若欲提升教学效果、落实“以学生为中心”的理念，并尊重和关注不同层次学生的发展差异，需实施分层教学。分层依据可涵盖学生知识基础、学习兴趣以及就业或考研意愿等维度，并根据学生学习表现与需求变化进行动态调整[4]。

2.3. 项目式教学研究

项目式教学侧重培养学生实践动手能力，推动理论知识向实践应用转化，具备突出的创新性与综合性特征，目前已在各类工科课程教学中得到广泛应用。Ngereja B 等学者[5]借助实证研究，验证了项目式教学能够有效激发学生的内部学习动机、提升其自主学习水平，相关研究结论可为项目式教学模式的改革落地提供实践层面的参考依据。程序设计课程教学效果不佳、学生课堂参与积极性不足的现状，相关研究融合 PBL 项目与 BOPPPS 构建混合教学模式，实践证实该模式能够增强学生学习主动性，显著提高课堂参与度与课程考核成绩[6]。

3. 生成式 AI 支持下 Python 课程分层支架教学模式设计

在多数 Python 课程教学中学生基础差异显著、对生成式 AI 太过于依赖弱化独立思考与调试能力、以及理论讲授与实践任务简单对应，易造成知识点碎片化，难以串联[7] [8]，阻碍学生实现代码进阶与项

目迁移。基于此，本研究构建“分层支架－AI 透明化－纠错共享－项目链进阶”四维联动教学模式，旨在引导学生将生成式 AI 作为学习辅助工具，在提高编程效率的同时发展自身编程能力，防止陷入对 AI 的被动依赖，确保学习者在人机协同中保持主体性与自主性。总体框架图(图 1)如下：

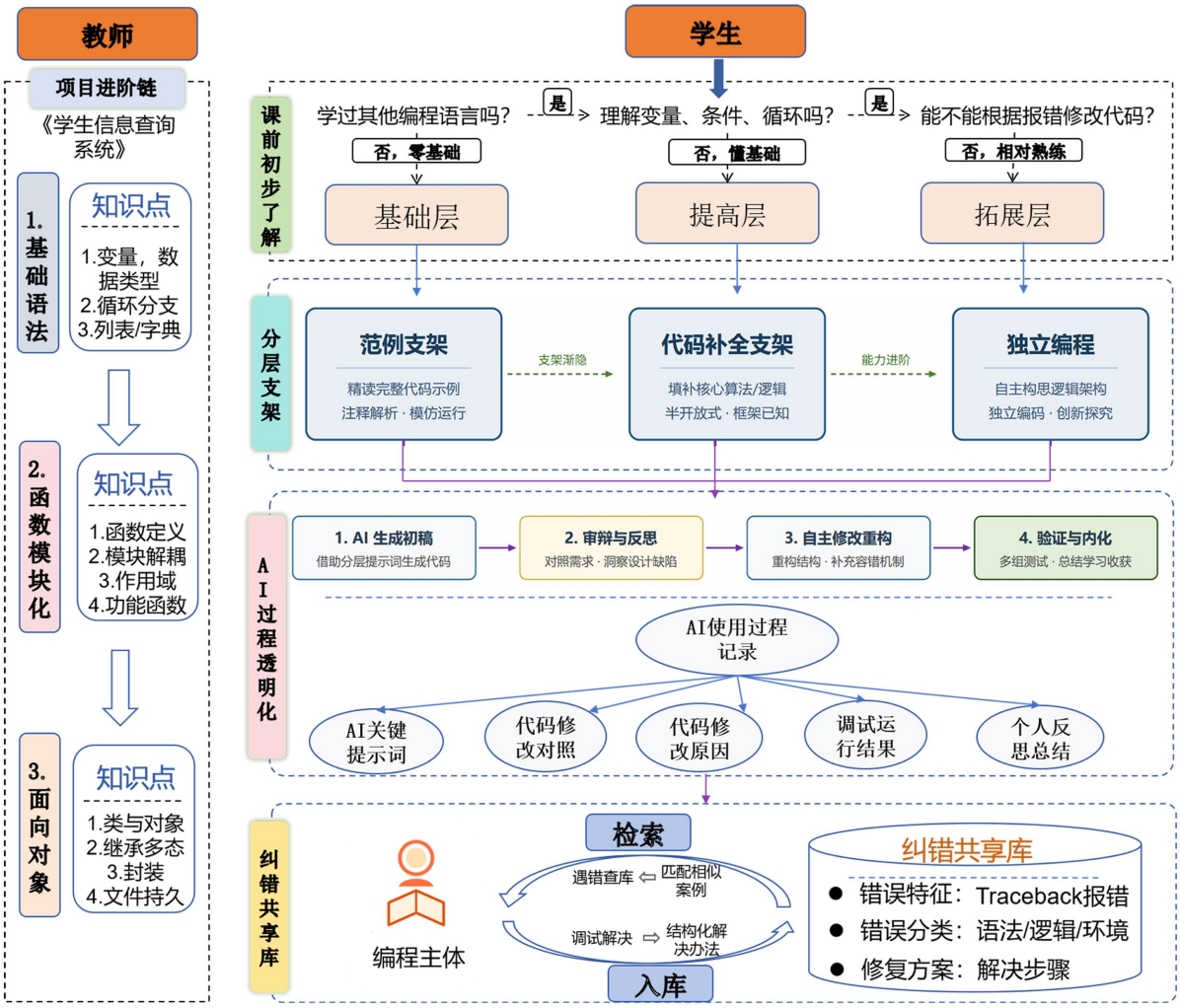


Figure 1. Diagram of the tiered framework for practical teaching
图 1. 分层支架实践教学体系图

3.1. 分层支架体系建立

为回应学生在编程基础、问题理解和项目实践方面的差异，本教学法以学习状态诊断为起点，将项目任务逐步推进，并据此提供不同程度的学习支持。课前，教师先通过简短问题了解学生是否接触过编程、能否理解变量和循环等基本概念，以及面对报错时能否尝试修改代码，诊断结果用于调整任务难度和支架强度，学生也可以根据后续表现进入更适合自己的学习路径。基础较弱的学生会获得更具体的帮助，例如语法说明、示例代码和操作提示，先完成项目的核心功能，对于具备一定基础的学生则减少过程性指导，更多地独立分析问题、拆分任务，并尝试用函数组织代码，对于基础扎实、探究意愿较强的学生，教师会增加代码优化和功能改进等要求，引导他们进一步理解程序结构，接触面向对象的设计思路。

此方法并不把学生长期固定在某一层级，教师会结合任务完成情况和学习表现，及时调整支架支持方式，学生由此逐步经历从理解知识、解决具体问题到迁移运用的过程，学习重点也会从“把程序做出来”转向“理解程序为什么这样设计”。

3.2. AI 过程透明化

本教学方法将生成式人工智能定位为编程学习中的辅助工具，强调学生需要进行过程记录、结果对比和反思总结，从而呈现学生使用 AI 的真实过程。当学生开展项目实践时，需要保存与 AI 互动的相关材料，逐步形成能够反映学习过程的记录。在项目开始阶段，学生可以借助 AI 生成程序初稿，同时保存自己提出的提示语，并说明这些提示与任务要求之间的关系；在有初步代码后，学生需要阅读程序结构，检查代码是否符合项目目标，当发现问题时需要记录判断依据并思考修改思路；经过分析，学生再结合自己的理解调整程序，通过版本对照呈现 AI 建议与个人修改之间的差异；修改完成后，学生运行程序并进行调试，记录报错原因、处理过程和最终结果。

借助该方式，AI 介入学习的完整链路能够清晰呈现。教师可据此掌握学生的分析、推理与知识迁移水平，学习重心也从直接获取代码，转变为读懂代码、验证运行结果，并结合实际需求迭代优化方案。

AI 辅助学习过程记录与评价

为避免学生将生成式人工智能工具简单作为代码生成器使用，本研究要求实验组学生完整记录 AI 辅助学习过程，并将过程记录纳入课程过程性评价。学生在完成编程任务时，应围绕“问题理解－独立分析－AI 交互－结果核验－代码修改－调试验证－反思迁移”这一过程链条提交学习记录。记录内容主要包括任务要求与问题描述、使用 AI 前的初步思考、AI 提示词及交互内容、AI 生成代码、个人修改过程、测试与调试结果以及学习反思等。

具体而言，学生首先需要说明任务目标、功能要求、输入输出条件、限制条件和当前遇到的具体问题；在使用 AI 工具之前，应记录个人已有的解决思路、初步代码或尝试结果。使用 AI 工具时，应保存具有代表性的提示词和对话内容，并说明提出该问题的原因。获得 AI 生成的代码或解决方案后，学生不能直接复制使用，而应结合运行结果、测试数据、程序逻辑和课程知识对 AI 结果进行分析和验证。对于经过修改的代码，应说明修改位置、修改原因以及修改是否解决了原有问题。任务完成后，学生还需总结本次任务中掌握的知识、AI 工具发挥的作用及其局限，并提出后续改进或迁移应用的计划。

为提高评价的可操作性，本研究制定“AI 辅助学习过程记录评价量规”，从问题描述与任务理解、独立分析与初步尝试、提示词设计与迭代、AI 结果理解、批判性分析与结果验证、代码修改合理性、调试过程完整性以及反思总结与知识迁移八个维度对学生的过程性表现进行评价。表 1 展示了评价量规，各维度采用四级评分制，分别对应“优秀”“良好”“合格”和“不合格”，并根据各维度权重计算过程记录总成绩。

Table 1. AI-assisted learning process documentation and evaluation rubric
表 1. AI 辅助学习过程记录评价量规

评价维度	权重	4 级：优秀	3 级：良好	2 级：合格	1 级：不合格
问题描述与任务理解	10%	能够准确说明任务目标、功能要求、输入输出、限制条件和具体困难，问题描述清晰完整，能够为后续分析提供明确依据	能够说明任务目标和主要困难，但个别条件或细节不够完整	能够概括基本任务，但问题描述较笼统，缺少部分关键条件	未能准确理解任务，或仅简单复制题目，无法说明具体问题

续表

独立分析与初步尝试	10%	在使用 AI 前记录了个人思路、相关知识依据、解决假设和初步代码，并说明尝试结果	有较明确的初步思路或代码尝试，但分析过程不够完整	有简单尝试，但缺少原因分析、过程说明或结果记录	未进行实质性分析和尝试，直接要求 AI 生成完整答案
提示词设计与迭代	15%	提示词包含任务背景、具体问题、约束条件和输出要求；能够根据 AI 反馈进行有目的的多轮调整，并主动要求解释、测试或比较方案	提示词与任务具有较强相关性，能够获得较有针对性的帮助，但背景信息或迭代过程略有不足	能够提出基本问题，但提示词较笼统，主要依赖一次性询问	提示词缺乏具体信息，主要使用“直接写代码”或“直接修改错误”等简单指令
对 AI 生成结果的理解	15%	能够用自己的语言解释代码结构、关键语句、函数作用、运行逻辑及适用条件，并指出 AI 方案的假设或局限	基本能够解释主要代码逻辑，但对部分细节或适用条件说明不足	只能说明代码的表面功能，对关键语句和运行机制理解不充分	无法解释 AI 生成内容，主要以“AI 给出的答案”或“代码能够运行”为依据
批判性分析与结果验证	20%	能够通过运行测试、边界条件检验、结果比对、资料查阅或方案比较对 AI 结果进行独立核验，并主动发现错误或不适用之处	完成了基本运行测试，能够识别较明显的问题并进行修正	仅对正常情况进行简单测试，验证范围有限，缺少对异常情况的分析	未进行有效验证，直接接受或复制 AI 生成结果
代码修改的合理性	15%	清楚呈现修改前后代码或版本差异，逐项说明修改原因；修改具有针对性、必要性和可解释性	能够说明主要修改内容和部分修改理由，代码修改总体合理	进行了代码修改，但部分修改原因不清楚，存在较明显的试错性	大量复制 AI 生成代码，无法说明修改内容及其原因
调试过程的完整性	10%	完整记录错误现象、错误原因、调试策略、修改过程和验证结果，能够根据错误信息逐步定位问题	记录了主要错误和修复过程，但对错误原因或验证过程说明不足	有错误记录和修改结果，但调试过程较为零散	缺少错误记录，或仅呈现最终可运行代码，无法说明调试过程
反思总结与知识迁移	15%	能够具体总结任务中的知识收获、AI 工具的作用与局限、个人不足及改进计划，并能将所学方法迁移到新的任务情境	能够总结主要收获和 AI 使用体验，并提出一定的改进方向	反思内容较为概括，主要停留在任务完成情况或工具使用感受层面	反思内容简单、空泛或与实际学习过程不符，未体现个人理解和后续计划

注：各维度按照 1~4 级评分。过程记录总分可按照“各维度得分 ÷ 4 × 该维度权重”进行计算，满分为 100 分。若学生未提交某一维度的有效材料，该维度原则上记为 0 分，并在评价记录中注明原因。

3.3. 纠错共享

为帮助学生积累编程调试经验并实现经验迁移，本课程在开课之初就搭建了一个由学生共同参与维护的纠错案例库，并贯穿教学全过程。由于课程项目任务统一、理论知识点相同，各类问题的解决思路具备良好的可推广性。在《学生信息查询系统》项目中，若遇到报错或功能异常时，先保存运行现象、报错提示和相关代码，再结合程序运行结果推测问题原因，调试完成后，筛选代表性案例进行整理，写明错误现象、问题分类、解决方案以及调试思考过程。案例经教师审核通过后纳入共享库，案例贡献者可获得课程积分奖励。纠错共享网站示意图如图 2 所示。

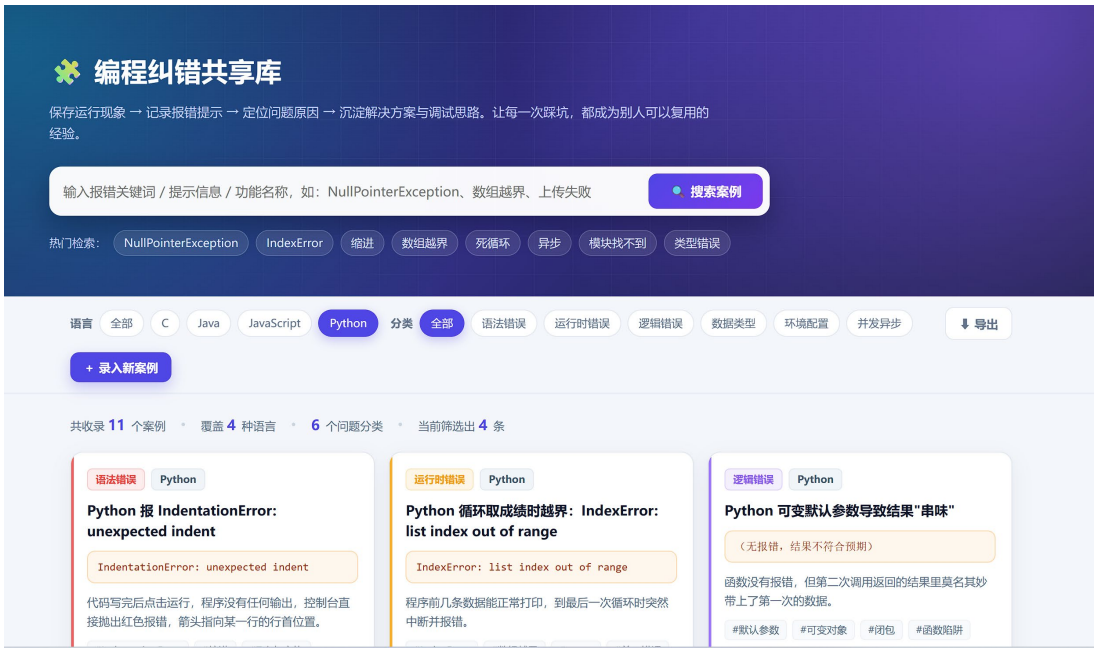


Figure 2. Diagram of an error-correction sharing website
图 2. 纠错共享网站示意图

当其他学生遇到问题时，可通过报错关键词、提示信息或功能名称检索案例，借鉴成熟的排查思路，减少盲目试错，案例库随教学推进持续补充完善，逐步形成一套动态更新的编程纠错题库，bug 案例详情图如图 3 所示。

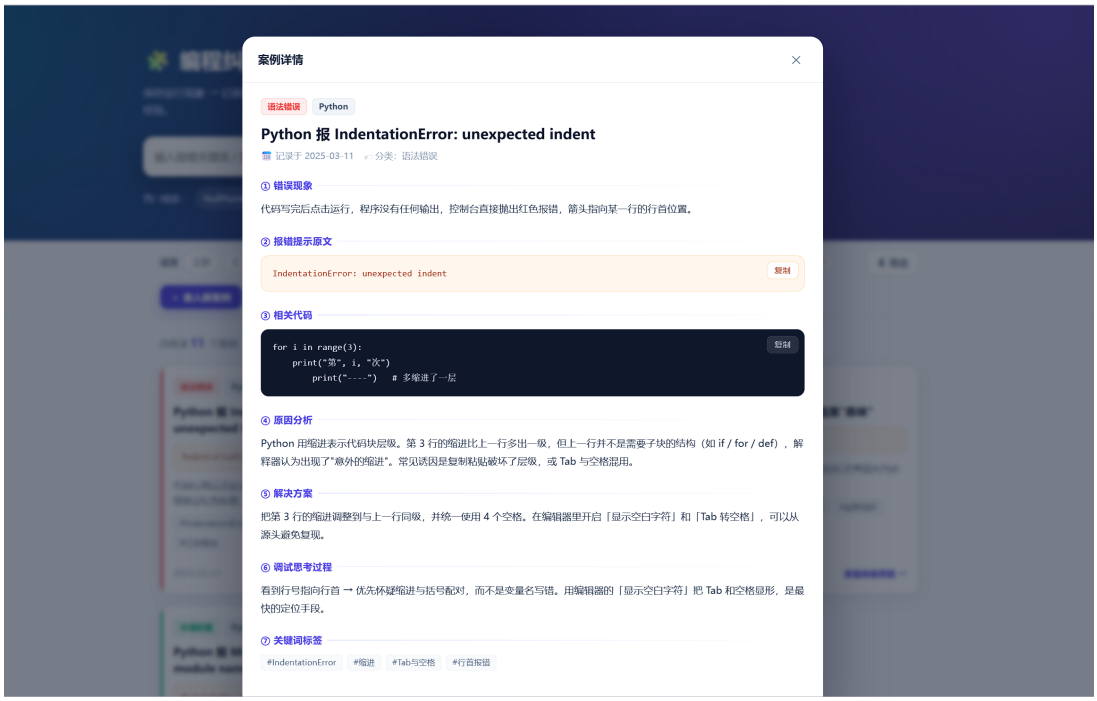


Figure 3. Detailed images of specific case studies from the error-correction and sharing website
图 3. 纠错共享网站具体案例详情图

3.4. 项目链进阶

在整个教学过程中，教师将以《学生信息查询系统》贯穿学习全过程，不再将变量、循环、函数等知识点割裂讲授，而是让学生以完善同一系统为目标逐步学习和使用这些知识。在项目初始阶段，学生先运用变量、数据类型、循环、分支和列表等基础语法，完成学生信息的录入、存储与简单查询，重点解决“程序能不能运行、功能能不能实现”的问题。随着任务的推进，系统需要增加按学号查询、按姓名检索和结果展示等功能，原有代码开始变得冗长，修改起来也不方便。此时，教师引导学生将不同功能拆分为独立函数，明确函数的参数、返回值和作用范围，逐步形成模块化编程意识。功能增加后，学生还会遇到数据与操作相互分离、代码重复以及维护困难等问题。教师借助这些真实的编程困难，引出类与对象、封装、继承和多态等面向对象知识，指导学生将学生信息及其相关操作进行重新组织，完成系统的面向对象重构。这样，知识学习始终服务于项目改进，项目也随着知识的深入逐步从“能够查询”发展为“结构清晰、便于扩展”的学生信息查询系统，学生的学习过程则由基础语法应用，逐渐过渡到函数模块化，再提升到程序结构设计与面向对象思维。

表 2 清晰呈现了《学生信息查询系统》项目的三阶进阶链路，将项目拆解为 V1.0 基础查询版、V2.0 模块化查询版、V3.0 面向对象版三个迭代版本，分别对应基础语法、函数模块化、面向对象三个递进学习阶段。表格把每个版本对应的核心知识点、需要落地实现的功能内容以及阶段培养目标一一对应，直观展现项目迭代与知识点学习同步推进的设计思路。学生在每一个版本开发中，都能带着明确的工程目标去学习 Python 知识，在解决项目真实问题的过程中完成知识内化；教师也可依托该进阶链规划分层教学任务，按版本设置阶段性验收标准，循序渐进引导学生从会写简单代码，成长为具备模块化思维与面向对象程序设计能力的开发者，实现“项目驱动、做中学”的教学理念落地。

Table 2. Student information inquiry system project advanced practice
表 2. 《学生信息查询系统》项目进阶链

项目版本	对应阶段	核心知识点	主要实现内容	阶段目标
V1.0 基础查询版	基础语法	变量与数据类型、输入与输出、条件分支、循环、列表与字典	使用列表和字典保存学生的学号、姓名、性别、成绩等信息；实现学生信息录入、按学号查询和结果输出等基本功能。	能够运用基础语法编写一个可以运行的学生信息查询程序，理解程序的基本执行流程。
V2.0 模块化查询版	函数模块化	函数定义与调用、参数传递、返回值、变量作用域、模块化设计	将学生信息录入、按学号查询、按姓名查询、按成绩筛选、信息展示等功能分别封装为独立函数；通过主函数统一调度各个功能模块。	减少重复代码，理清程序结构，初步形成按照功能拆分问题和组织代码的意识。
V3.0 面向对象版	面向对象	类与对象、构造方法、属性与方法、封装、继承、多态、文件读写	创建 Student 学生类，将学生的基本信息和相关操作进行封装；根据学生类型设计本科生、研究生等子类；实现学生数据的文件保存与读取，并完善查询和管理功能。	完成系统的结构化重构，提高程序的可维护性和可扩展性，理解面向对象的程序设计。

4. 实践成效与反思

4.1. 实践成效

本教学改进方案在四川大学锦江学院计算机学院 2023 级软件工程专业《Python 程序设计》课程开展

为期一学期的教学实践，课程总学时 64 学时，包含理论 32 学时、实践 32 学时，采用每周理论 + 实践分段授课模式，依托 PyCharm 本地 IDE 开展编程训练。经过一学期教学落地，取得如下成效：

1) 学生课程参与积极性显著提高。依托学生共建纠错案例库机制与课程积分激励，学生从被动听课、完成任务转变为主动记录调试案例、梳理报错思路，学生愿意主动分享自己遇到的程序故障与解决经验，参与案例撰写、补充修订的热情较高，形成互助研讨的课堂氛围。

2) 学生编程调试与问题迁移能力明显增强。借助共享纠错案例库，学生遇到报错时可检索同类案例，借鉴排查思路，减少盲目试错，在《学生信息查询系统》项目开发过程中，学生能够自主识别错误类型、分析问题根源，并且学会将已掌握的调试经验迁移至新的编程场景，自主排查和修复代码异常的能力得到锻炼。

3) 降低学生对 AI 工具的依赖，建立编程自信心。教学中将学习重心由直接获取代码转向读懂代码、检验结果与迭代改进，学生不再直接借助 AI 生成完整代码，而是自主分析需求、动手编写并独立调试程序。在持续的排错实践中，学生逐步建立起解决编程问题的信心，更加相信自身的代码分析与问题处理能力。

4.2. 研究设计与效果评价

为进一步验证“分层支架 - AI 透明化 - 纠错共享 - 项目链进阶”教学模式的实际效果，本研究拟采用准实验研究设计，选取《Python 程序设计》课程中的两个平行班作为研究对象。在教学内容、课时安排、授课教师、课程考核要求等条件基本一致的前提下，将其中一个班设置为实验组，采用生成式 AI 支持下的分层支架教学模式；另一个班设置为对照组，采用传统教学模式，或仅保留部分教学干预措施。

实验组围绕“分层支架、AI 透明化、纠错共享和项目链进阶”开展教学。教师根据学生的编程基础和学习表现提供差异化任务与学习支架，要求学生记录 AI 使用过程、核验 AI 生成代码、提交调试过程和反思材料，并通过递进式项目促进知识迁移。对照组则按照原有教学方案开展教学，主要采用教师讲授、教材练习和常规项目实践等方式；如设置部分干预对照组，可仅实施项目式教学或允许使用生成式 AI，但不要求提交 AI 交互记录、错误案例和调试反思材料。

为全面评价教学模式的实施效果，本研究从学习结果、编程过程和学习心理三个层面收集数据。第一，收集学生的课程成绩、单元测试成绩、编程任务完成情况和综合项目成绩，以评价学生的知识掌握和综合应用能力。第二，邀请具有相关教学或程序设计经验的教师依据统一评分量表，对学生代码的正确性、规范性、可读性、模块化程度、可维护性和扩展性进行评价，以分析不同教学模式对代码质量的影响。第三，通过前测和后测问卷调查学生的学习动机、编程自我效能感、学习投入程度、AI 使用态度以及 AI 依赖倾向等变化。第四，记录学生在课程平台、代码提交系统和 AI 学习工具中的使用行为，包括代码提交次数、调试次数、代码修改次数、AI 交互频率、提示语类型、AI 生成代码的使用比例、代码解释与验证行为等，以分析学生的学习过程和 AI 使用特征。必要时，还可选取部分学生和教师开展半结构式访谈，以进一步了解教学模式的实施体验、困难及其作用机制。

在数据分析方面，首先采用描述性统计方法呈现两组学生在各项指标上的均值、标准差和变化趋势；其次，根据数据分布情况，采用独立样本检验、配对样本检验或协方差分析比较两组学生的学习效果差异，并结合效应量评估教学模式的实际影响程度。对于前测成绩存在差异的情况，可将前测成绩作为协变量，以提高组间比较的准确性。同时，可进一步分析学生编程基础、AI 使用频率与教学效果之间的关系，探讨该教学模式对不同基础层次学生的差异化影响。对于访谈资料、AI 使用记录和调试反思材料，则采用主题分析或内容分析方法，归纳学生在问题分析、代码理解、错误定位、知识迁移和 AI 依赖等方面的表现变化。

通过定量数据与定性资料的交叉验证，可以从学习成绩、代码质量、学习心理、学习行为和学习体验等多个维度考察该教学模式的有效性。与仅依据课程成绩或学生主观反馈进行评价相比，该研究设计能够更全面地揭示教学模式对学生编程能力、自主学习能力和生成式 AI 使用方式的影响，为其教学成效提供更加客观、可靠的实证依据。

4.3. 不足与反思

本教学方法采用“分层支架－AI 透明化－纠错共享－项目链进阶”四维联动教学模式，在取得一定成果的同时，但仍然在实施过程中遇到了一些问题，需要在今后的教学过程中进行改进。

第一，案例库质量参差不齐，筛选与维护成本较高。案例由学生自主提交，部分学生撰写的案例描述简略，错误原因分析流于表面，缺少对底层原理的剖析；少数案例存在表述不规范、解决方案通用性差等问题。教师需要花费较多时间审核、校对内容。

第二，生成式 AI 的使用缺乏可落地的管控与引导标准。本教学改革目标是引导学生合理运用生成式 AI 辅助学习，但尚未建立清晰的使用规范与分层引导方案。部分学生对 AI 工具的使用边界把握不清，仍存在过度依赖 AI 生成代码、直接复制解决方案的现象，难以区分借助 AI 查阅知识点和完全交由 AI 完成程序开发的差异，削弱了自主思考与代码调试训练的效果。

5. 结语

针对 Python 课程教学中存在的现实痛点：生源基础差距较大、学生过度依赖生成式 AI 导致独立思考与代码调试能力弱化，同时理论教学与实践任务浅层次匹配，知识碎片化，不利于学生代码能力进阶与知识迁移。本研究构建“分层支架－AI 透明化－纠错共享－项目链进阶”四维联动教学模式。该模式以人机协同学习为核心，引导学生将生成式 AI 定位为学习辅助工具，在发挥 AI 提升学习效率优势的前提下，减少学生对 AI 的被动依赖，保障学生在编程学习过程中的主体地位，锻炼自主分析、代码调试与知识迁移能力，为 Python 程序设计课程的实践教学改革提供可行思路。

参考文献

[1] 张健. “师-生-机”三元交互: AI 赋能下的数值分析课程设计教学模式创新研究[J]. 创新教育研究, 2025, 13(11): 292-299.

[2] 王艳芝, 张春莉. 学习支架何以促学——基于皮亚杰与维果茨基思想的综合视角[J]. 教育科学研究, 2024(11): 76-82.

[3] 陈豪, 吴航, 高宇航, 朱庆香. 基于支架式教学的 Python 编程实验教学改革与实践[J]. 社会科学前沿, 2026, 15(7): 226-232.

[4] 朱应丽. 混合生源分班制下分层教学模式在高职数学教学中的应用分析[J]. 职业教育发展, 2021, 10(3): 176-179.

[5] Ngereja, B., Hussein, B. and Andersen, B. (2020) Does Project-Based Learning (PBL) Promote Student Learning? A Performance Evaluation. *Education Sciences*, **10**, Article No. 330. https://doi.org/10.3390/educsci10110330

[6] 刘忠慧, 覃利华. BOPPPS + PBL 混合教学模式在《Python 程序设计》教学中的应用探究[J]. 教育进展, 2026, 16(3): 223-231.

[7] Agustina, N. (2016) Assessment of Project-Based Learning in Science Class. *Jurnal Siliwangi*: *Seri Pendidikan*, **2**, 137-150.

[8] Wit, E. and McClure, J. (2004) Statistics for Microarrays: Design, Analysis and Inference. 2nd Edition. John Wiley & Sons. https://doi.org/10.1002/0470011084