

生成式人工智能背景下《Python语言程序设计》教学模式探索与实践

马国财*, 李晓婉, 黄雅文, 张怀聪

青海理工学院计算机与信息科学学院, 青海 西宁

收稿日期: 2026年9月12日; 录用日期: 2026年10月14日; 发布日期: 2026年10月20日

摘 要

随着生成式人工智能技术的快速发展, DeepSeek、豆包等AI工具逐渐应用于程序设计课程学习过程。虽然AI工具能够提高学习效率, 但部分学生在学习过程中存在过度依赖现象, 导致独立思考能力、程序设计能力以及创新意识培养受到一定影响。针对这一问题, 本文以《Python语言程序设计》课程为研究对象, 结合大一新生学习特点, 构建了一种以培养学生独立思考能力为核心的教学模式。该模式通过原创编程任务驱动、翻转课堂以及阶段性编程竞赛等教学活动, 引导学生主动进行问题分析、程序设计和程序实现的完整学习过程。经过一个学期的教学实践, 学生课堂参与度、自主学习意识以及程序设计能力均得到明显提升, 对生成式AI工具的使用也更加理性。该教学模式能够在发挥AI辅助学习优势的同时, 有效避免学生形成过度依赖, 为生成式AI背景下程序设计类课程教学改革提供了参考。

关键词

生成式人工智能, 翻转课堂, Python语言程序设计, 自主学习, 程序设计能力

Exploration and Practice of Teaching Models for “Python Language Programming” in the Context of Generative AI

Guocai Ma*, Xiaowan Li, Yawen Huang, Huaicong Zhang

Qinghai Institute of Technology, School of Computer and Information Sciences, Xining Qinghai

Received: September 12, 2026; accepted: October 14, 2026; published: October 20, 2026

*通讯作者。

文章引用: 马国财, 李晓婉, 黄雅文, 张怀聪. 生成式人工智能背景下《Python 语言程序设计》教学模式探索与实践[J]. 教育进展, 2026, 16(10): 1219-1225. DOI: 10.12677/ae.2026.16102222

Abstract

With the rapid advancement of generative artificial intelligence technologies, AI tools such as DeepSeek and Doubao have been increasingly applied in the learning process of programming courses. Although these AI tools can improve learning efficiency, some students have developed an over-reliance on them, which adversely affects the cultivation of independent thinking ability, programming proficiency, and innovative awareness. To address this issue, this study takes the *Python Language Programming* course as the research subject and, taking into account the learning characteristics of first-year college students, constructs a teaching model centered on fostering students' independent thinking ability. This model integrates original programming task-driven learning, flipped classroom, and phased programming competitions to guide students through the complete learning process of problem analysis, program design, and program implementation. After one semester of teaching practice, students' classroom participation, autonomous learning awareness, and programming abilities have been significantly improved, and their use of generative AI tools has become more rational. This teaching model effectively prevents students from developing excessive dependence on AI while leveraging its advantages in assisting learning, thereby providing a reference for the reform of programming course instruction in the context of generative AI.

Keywords

Generative Artificial Intelligence, Flipped Classroom, Python Language Programming, Autonomous Learning, Programming Ability

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

近年来,生成式人工智能(Generative AI)的快速发展在教育领域被广泛应用,知识的获取方式多样化。西门子提出的联通主义认为,数字时代的知识呈分布式网络形态,学习的核心是建立有效连接。传统的编程语言类教学如《Python 语言程序设计》中,教学侧重学生对基本语法知识的掌握和应用,而如今,生成式人工智能工具的普遍使用,学生学习模式发生巨大变化,然而这促使此类课程的教学模式也发生变化。

《Python 语言程序设计》[1]作为大一新生入门的编程语言课程,帮助学生构建编程思想,培养学生良好的编码习惯,让学生掌握基本的编程语言语法知识。如果学生在学习之初,严重依赖生成式 AI 工具,放弃主动学习思考,放弃主动检查代码 bug 的机会,学习效果将大大折扣,这与我们的教学目标将背道而驰。

基于此,本文受同伴互助学习理论[2]、翻转课堂理论[3]、元认知理论的启发[4][5],在生成式 AI 背景下,提出“防依赖-促独立”的教学策略,主要包括:自主思考-自主实践-自主表达-自主创新,培养学生主动学习的能力,帮助学生构建编程思想,从而实现编程学习的深度和广度上的提升。

2. 生成式 AI 背景下《Python 语言程序设计》课程教学面临的挑战

2.1. AI 依赖

随着生成式人工智能工具的快速发展与迭代,学生的编程成本大幅度降低,从来自行掌握语法知识、理解题目、编写代码、修改错误、调整优化,到现在的“一键获取”,不管是课前、课中还是课后,

学生都严重依赖 AI 工具[6]。作业让 AI 写,从表面来看,学生能够更快地提交作业,完成任务,但从根本上来看,作业和思考任务并没有达到教师预期的教学目标。学生放弃了主动探索的乐趣和修改好 bug 后获得的成就感。学生难以感受到学习带来的收获的喜悦和思考的全过程。

2.2. 独立思考能力弱化

学习的过程是思维不断构建的过程,知识带来的不仅是知识本身,而是在获取知识的过程中产生的思考,当学生严重依靠 AI 工具完成学习任务时,如借助不当,将弱化或放弃个人思考过程[7]。其带来的后果是严重的,尤其对于还未接触过或少量接触编程的学生。它会导致学生的惰性思考,这将会恶性循环,当出现复杂思考问题时,借助生成式工具完成,逐渐地,简单地思考问题也会编程“懒得思考”。影响学生的学习习惯,从而会影响学生后续相关课程的学习。

2.3. 编程基础知识掌握不够扎实

《Python 语言程序设计》课程具有较强的基础性和衔接性。变量、数据类型、流程控制、函数设计以及面向对象程序设计等知识之间层层递进,任何一个环节掌握不牢都可能影响后续学习效果。该课程的学习过程更是尤为重要,尤其是对课后布置的任务,更需要自己耐心拆解问题,熟练运用所学知识点,编写代码,反复调试,才能够熟练掌握。其中,如学生过多依赖生成式 AI 工具,尤其是代码调试和运行阶段,学生无法发现问题,无法运用自己所学解决问题,学生对基本知识得不到有效运用,会导致学生原本就不太夯实的基础更加薄弱[8]。

2.4. 学习主动性与创新意识有所弱化

在《Python 语言程序设计》课程的学习过程中,学生可以通过自己不断调试、不断思考、不断检查代码,最终解决问题、调试成功中获得正向激励,会让学生有改好 bug 后的成就感,从而调动学生主动学习,主动探索,主动解决问题。众所周知,个人的创造力对于学生的发展是十分重要的。

然而,生成式 AI 给出的解决方案通常具有较高规范性,但不同学生获得的答案往往趋于一致,容易导致解题思路同质化。在这种情况下,学生个性化思维的发展空间受到一定限制,不利于创新意识和创造性解决问题能力的培养。

2.5. 教学改革需求分析

综上,生成式人工智能虽然为课程教学提供了便利条件,但同时也给学生培养带来了挑战。对于《Python 语言程序设计》课程而言,其教学目标不仅是学生掌握基本语法知识,完成代码调试,更重要的是培养学生独立思考能力、解决问题能力、创新能力、实践动手能力。因此,课程教学改革应该通过科学的教学设计,引导学生以正确方式使用 AI 工具,帮助学生回到问题本身,善于思考,勤于动手。

本文结合课程特点与学生实际情况,构建了以培养学生独立思考能力为核心的《Python 语言程序设计》教学模式,通过原创编程任务驱动、学生课堂讲解以及阶段性编程竞赛等教学活动,促进学生自主学习能力和程序设计能力的提升。

3. 核心机制

针对第 2 章中提到的问题,本文以《Python 语言程序设计》为例设计了“防依赖-促独立”的教学策略,通过原创编程任务驱动、同伴伴学机制以及阶段性编程竞赛等教学活动,培养学生的自主自考、自主实践、自主表达、自主创新的能力。该方法主要包括图 1 所示的四部分组成,总体设计思路如下图 1 所示,各环节相互衔接,依次增强,在不同阶段逐级培养学生的多维度能力。

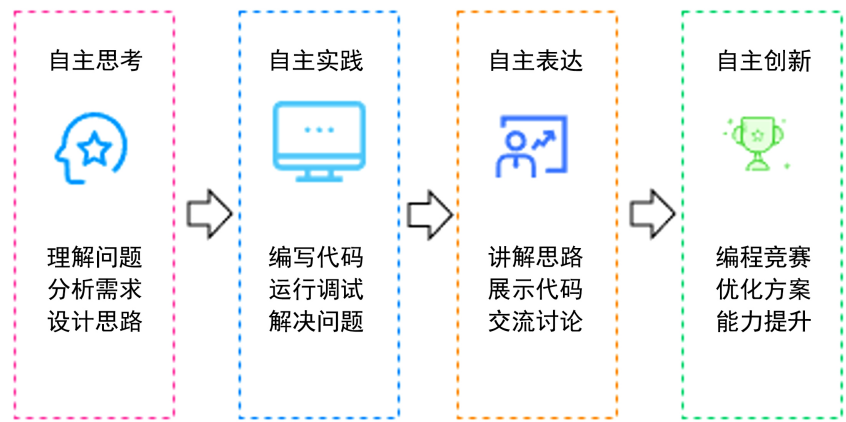


Figure 1. Overall framework diagram of teaching mode
图 1. 教学模式总体框架图

3.1. 编程任务驱动的自主学习机制

针对学生容易直接利用 AI 获取答案的问题，教师在具体的教学环节中，每次课堂结束前，围绕本节课程所学知识布置相应的练习任务，结合学生兴趣，特点，专业背景设置，例如环境科学班级可设置类似“水质评价指数计算，空气质量等级判定”等设计学生练习任务。在任务提交要求上，要求学生不仅需要提交代码，还需要提供思考的过程，提交后的作业教师认真批改，挑选优秀代码供下次课堂代码分享，也即下一小结 3.2 同伴互学机制。

这一设计的主要目的是让学生通过自主分析问题，亲自编写代码，逐步减少对生成式 AI 工具的依赖，在解决问题的过程中，运用知识点并将方案付诸实践，最终掌握知识点。

3.2. 学生课堂讲解与同伴互学机制

该环节设计出发点为两点，第一在日常教学环节中，我们发现，学生面对上台发言和分享是胆怯的，即使代码写得很好，并且很想上台分享，但内心也缺乏上台的勇气，因此，在实际教学环节中我们加入学生课堂讲解的方法，锻炼学生的语言表达能力；第二上台讲解代码能够让讲解的学生进一步加强知识点掌握，提高编写代码的能力，梳理思路，对于台下的学生而言，同伴讲解更能吸引学生的注意力。根据同伴互助学习理论(Peer Tutoring, Vygotsky 等)同伴间的解释、讨论、互评能够促进深度学习。

该环节具体机制是双向的，教师可通过布置的任务、作业情况掌握学生对每一节知识点的掌握情况，根据学生的完成情况，教师可挑选代码编写优秀的学生作为下次课余时间分享的候选人，以便于让班级更多学生学习他们良好的编码习惯和分析问题解决问题的思路。其次，课堂上学生也可根据个人情况提前自愿报名、准备、讲解，凡上台分享的学生均有平时成绩的奖励。该过程为翻转课堂课外自学与课内消化两个环节，通过翻转课堂可以调用更多的学生参与本课程的学习，课下更多地思考问题、解决问题、编写代码、课上还可以分享自己的思路，代码还可以获得平时成绩的奖励，让学生在翻转课堂中获得更多学习该课程的乐趣。



Figure 2. Schematic diagram of the student classroom explanation process
图 2. 学生课堂讲解流程示意图

该环节如图 2 所示, 主要包括以下步骤: 学生自愿报名、学生选择讲解题目、学生课后准备、学生上台讲解、同学之间的交流讨论、教师点评总结、反思改进方案。各步骤间紧密衔接, 共同构成符合本课程的翻转课堂实践实例。

3.3. 阶段性编程竞赛激励机制

学生在关键模块、学期中、学期末, 在掌握或基本掌握 Python 语言程序设计基本知识点后, 展开阶段性编程比赛, 组织班级学生积极参与, 阶段性编程竞赛主要包括如下图 3 所示流程: 确定竞赛主题、组织竞赛报名、发布竞赛题目、学生独立完成、评审与展示、结果反馈与奖励。确定竞赛主题可由任课教师或对应课程组老师共同设计, 题目的考察点和难易程度应结合班级学生特点, 设计适合学生的竞赛题目。第二环节为组织竞赛报名, 积极鼓励学生参与, 并设计相应的激励机制, 提高学生的参与度。第三和后续环节竞赛环节, 后续环节保证学生公平公正地完成比赛, 教师对获奖同学和参与同学进行奖励, 总结并反馈给学生。



Figure 3. Schematic diagram of student stage competition process

图 3. 学生阶段性竞赛流程示意图

4. 教学实施过程

4.1. 实施对象与周期

实践对象为 2025 级环境科学大一两个班, 共计 87 人, 其中男生 42 人, 女生 45 人。学生具有零基础或少量编程经验, 认知水平参差不齐。教学周期为一学期, 共 12 周, 每周安排 4 课时, 分别包括 2 课时理论教学和 2 课时实践教学, 课程内容由浅入深, 逐步覆盖 Python 基础知识、算法设计与程序开发能力的培养。

4.2. 课程模块安排

课程内容分为四大模块:

(1) Python 基本语法元素: 包括但不限于计算机概念、组成与计算机程序, 开发环境配置, Python 语言基本语法知识。

(2) 基本数据类型与程序控制结构: 主要包括数字类型, 类型转化与运算, 字符串类型及其基本操作, 程序的顺序结构, 分支结构, 循环结构, 异常处理。

(3) 函数与代码复用: 函数的定义, 调用, 递归函数的设计, 模块化设计。

(4) 组合数据类型: 主要包括集合、列表、元组、字典四种组合数据类型的讲解。

(5) 文件处理与基本数据分析: 主要讲解文件读写, 数据格式化, 简单基础的数据分析与可视化。

4.3. 教学环节实施

4.3.1. 课前原创编程任务

每周理论教学课堂中会讲解新的理论知识点, 教师根据每节课理论教学知识点情况和学生在课堂上的掌握情况和反馈, 设计当次课后编程任务。要求学生可以不完全编写并调试成功, 根据自己掌握的情况, 思考, 编写。要求提交的代码必须是自己亲手写出的, 并体现思考过程, 做好注释, 以便于教师检

查。该阶段学生可根据教师布置的任务，自行挑选任务在下次课程中进行同伴伴学课堂讲解，教师可给予平时成绩奖励。

4.3.2. 课堂讲解与同伴互学

每节实践或理论课程教师会预留 5~10 分钟时间，将课堂“交给”学生。学生可自发报名或教师在学生提交的代码中挑选编写较好的学生上台分享。分享时学生不能拿着已编写好的代码直接展示，由于对于大一新生本课程布置的课后任务多以巩固基本知识为主，故编程题目较为简单，所以分享学生上台时需要从 0 开始一步一步编写，这样的好处是，能够体现学生的思考过程，台下的学生也能够更好的参与进来，帮助台上学生找出 bug，调试代码，能够让全班学生积极参与进来。

4.3.3. 阶段性编程竞赛

在每个关键模块、学期中或学期末，举办阶段性编程大赛，让学生在多种多样的竞赛题目中对课程内容有更深刻的掌握。在设计竞赛题目时，题目的难度居中，多以中等题为主，题目类型为编程题，考察学生对基本知识点的掌握情况和熟练运用能力，少量还有考察学生思维能力。学生的参赛题目构成题库，所有参赛的学生随机地从中抽取其中三道题目，参赛时长为 45 分钟。在规定时间内调试运行成功并结果正确即可得分，得分最高者获胜，我们根据学生参与人数，按比例设置奖励，一等奖 10%，少于 1 人时为 1 人，二等奖 20%，少于 2 人时为 2 人，三等奖 30% 少于 3 人时为 3 人。竞赛全过程独立完成，不得联网使用互联网工具、不得交头接耳。

4.3.4. AI 使用规范设计

为防止学生过度依赖生成式 AI，教学过程中明确以下规范：

- 1、学生完成原创任务时，应先独立思考，不允许直接使用 AI 生成完整代码。
- 2、AI 可作为调试工具或提供优化建议，但必须在独立完成任务的基础上使用。
- 3、学生需在课堂讲解或作业提交中说明 AI 辅助使用的环节，以便教师监督和指导。

通过合理引导，学生能够在自主思考和实践过程中逐步形成程序设计能力，同时学会科学利用 AI 工具辅助学习。

5. 教学效果分析

5.1. 学生独立编程能力明显提升

独立学习能力是学生需要掌握的核心能力之一，对于学生未来发展至关重要。通过一个学期的实践，从实践情况来看，部分学生在面对稍有难度的编程题目时容易产生畏难情绪，习惯向教师或 AI 工具寻求帮助。但随着原创任务训练次数增加，学生开始尝试自主分析题目要求，并能够利用所学知识完成程序设计。尤其是在循环结构、函数设计以及综合程序设计等模块学习过程中，学生独立完成程序任务的能力得到明显提升。

5.2. 学习主动性与课堂参与度显著增强

在本课程初期，课堂学生主动分享的人数较少，但通过学生自主选择题目、课下思考理解、代码编写、上台分享，学生逐步从编程中获得成就感，这促使学生更加愿意分享和课堂中回答问题。随着分享的学生越来越多，更多学生参与进来，翻转课堂，学生对于学习编程的主动性和积极性也逐渐提高。

5.3. 学生创新意识与学习兴趣得到激发

兴趣是最好的老师，能够促使学生自发地学习、探索。在本研究中，阶段性编程竞赛为学生激发学

习热情。从竞赛开展情况来看,大部分学生表现出较高参与热情。部分学生能够在完成基本功能的基础上主动增加扩展功能,体现出较强的创新意识和探索精神。一些学生还尝试采用不同算法实现相同功能,并在赛后与同学分享优化思路。弗拉维尔的元认知理论提醒我们,高效学习者需具备对自身思维过程的觉察与调控能力,所以学生在赛后反思与优化对学生成长至关重要。

5.4. 学生对生成式 AI 工具的使用更加理性

本研究并未简单限制学生使用生成式 AI 工具,而是通过教学设计引导学生正确认识 AI 工具的作用。

实施初期,部分学生倾向于直接获取完整程序代码;而在教学模式运行一段时间后,学生更多将 AI 作为辅助学习工具,用于检查程序错误、验证解决思路或优化代码结构。这种变化表明学生逐渐认识到程序设计能力的形成离不开自主思考和实践训练, AI 工具只能作为学习过程中的辅助资源,而不能替代个人能力的发展过程。

6. 结语

生成式 AI 的发展为程序设计课程教学带来了新的机遇与挑战。针对《Python 语言程序设计》课程中学生过度依赖 AI 的问题,本文构建了以原创编程任务、课堂讲解分享和阶段性编程竞赛为主要内容的教学模式,引导学生回归自主思考和实践探索过程。教学实践表明,该模式在提升学生独立编程能力、学习主动性和课堂参与度等方面取得了较好效果。

未来,课程教学将进一步探索生成式 AI 与程序设计教育的深度融合,充分发挥 AI 工具的辅助作用,同时持续强化学生自主学习能力和创新能力培养,为高校程序设计课程教学改革提供参考。

基金项目

青海省省级线上线下一流本科课程建设项目:《Python 语言程序设计》(项目编号:S-YLKC2025-1 XSXX02)。

参考文献

- [1] 嵩天,黄天羽. Python 语言程序设计教学案例新思维[J]. 计算机教育, 2017(12): 11-14, 19.
- [2] 田园. 基于活动理论的实训课同伴互助学习模式构建与实践[J]. 创新教育研究, 2025, 13(4): 204-213.
- [3] 陈晓菲. 翻转课堂教学模式的研究[D]: [硕士学位论文]. 武汉: 华中师范大学, 2014.
- [4] 周湘衡. 元认知理论及对教育研究的启示[J]. 当代教育科学, 1995(6): 11-12.
- [5] 姜英杰. 元认知研究的历史源流与发展趋势[J]. 东北师大学报(哲学社会科学版), 2007(2): 156-161.
- [6] 刘蕾, 张新亚. 人工智能依赖对创造力的影响与未来教育发展路径的省思[J]. 广西师范大学学报(哲学社会科学版), 2024, 60(1): 83-91.
- [7] 章剑和. 现代教育要努力培养学生的创造性思维品质[J]. 中国教育学刊, 1999(4): 17-20.
- [8] 孙丹, 朱城聪, 许作栋, 等. 基于生成式人工智能的大学生编程学习行为分析研究[J]. 电化教育研究, 2024, 45(3): 113-120.