《计算机与教育：人工智能》10 (2026) 100526

目录列表见ScienceDirect《计算机与教育：人工智能》期刊主页：

www.sciencedirect.com/journal/computers-and-education-artificial-intelligence

评估集成知识图谱的编程学习系统中的自适应与生成式人工智能反馈和推荐

Lalita Na Nongkhai [a]，Jingyun Wang [b, *]，Adam Wynn [b]，Takahiko Mendori [a]

[a]日本高知高知工科大学工程研究生院[b] 英国杜伦杜伦大学计算机科学系

ARTICLEINFO

关键词：自适应学习 推荐系统 大语言模型 检索增强生成 多种编程语言 编程教育

摘 要

本文介绍了一个框架的设计与开发，该框架将大语言模型（LLM）与检索增强生成（RAG）方法相结合，并同时利用知识图谱和用户交互历史。该框架被集成到一个此前开发的自适应学习支持系统中，用于评估学习者代码、生成形成性反馈并推荐练习。此外，本研究考察了学习者在三种教学模式下的偏好：自适应模式、生成式人工智能（GenAI）模式以及生成式人工智能—自适应混合模式。研究开展了一项实验研究，通过利用从所有实验组的4956次代码提交中提取的四项关键日志特征，比较学习者的学习表现与感知，以及这三种模式的有效性。分析结果表明，与接受自适应模式反馈的学习者相比，接受生成式人工智能模式反馈的学习者提交的正确代码显著更多，缺少基本编程逻辑的代码提交则更少。特别是，生成式人工智能与自适应混合模式取得了最多的正确提交次数，以及最少的错误或不完整尝试，其表现优于仅采用自适应模式和仅采用生成式人工智能模式的两组。问卷回答进一步表明，生成式人工智能生成的反馈被广泛认为是有帮助的，而所有模式在易用性和有用性方面均获得了积极评价。这些结果表明，在所有测量的日志特征方面，生成式人工智能与自适应混合模式均优于另外两种模式。

1. 引言

计算机编程教育长期以来一直面临挑战。初学者在理解编程概念、构建逻辑解决方案以及将这些解决方案转化为代码方面，常常遇到困难（Derus & Ali, 2012）。为应对这些挑战，有研究提出根据学习者个人表现推荐个性化学习路径并提供定制化内容（Chookaew et al., 2014; Gavrilovi´c et al., 2018）。尽管事实证明，持续投入独立的问题解决任务有助于提升编程技能（Djambong & Freiman, 2016），许多研究提出，应根据学习者的表现或认知水平提供自适应编程练习（Fabic et al., 2018; Zheng et al., 2022）。

除了个性化的编程实践之外，及时提供针对编程代码的形成性反馈也至关重要。根据Shute (2008)的定义，反馈是指旨在引导学习者反思学习、帮助他们消除错误观念或强化积极行为的信息。此外，Garcia（2025）强调，及时反馈是支架式教学中促进有效学习的关键要素，在编程教育中尤其如此。然而，学习者在线独立练习编程时，往往缺乏形成性反馈，只能得到正确与否的结果或错误消息。

为了提供详细反馈，许多研究人员探索了采用各种人工智能（AI）技术的方式（Deeva 等人，2021年；Frankford 等人，2025年）。近年来，作为专门针对语言任务进行优化的一类机器学习模型，大语言模型（LLM）迅速发展并得到广泛应用，尤其是在生成式人工智能（GenAI）领域，这也促使人们对其在编程教育中的应用产生了日益浓厚的兴趣。新兴研究已开始探索如何利用大语言模型来帮助

* 通讯作者。电子邮箱：lalita.nanongkhai88@gmail.com (L. Na Nongkhai), jingyun.wang@durham.ac.uk(J. Wang), adam.t.wynn@durham.ac.uk (A. Wynn), mendori.takahiko@kochi-tech.ac.jp (T. Mendori).

https://doi.org/10.1016/j.caeai.2025.100526收稿日期：2025年8月21日；修订稿收稿日期：2025年12月2日；接受日期：2025年12月4日
在线发布于2025年12月4日 2666-920X/© 2025 作者。由爱思唯尔有限公司出版。本文为开放获取文章，采用知识共享署名—非商业性使用—禁止演绎许可协议（http://creativecommons.org/licenses/by- nc-nd/4.0/）。

完成作业、生成代码解释、提供调试协助并给出个性化形成性反馈（Kosar 等人，2024年；George & Dewan，2024年；Majumdar 等人，2024年）。此外，研究人员还引入了检索增强生成（RAG）技术，通过整合代码库或知识图谱等背景信息或领域专属知识，提高人工智能生成的反馈的相关性和准确性（Li 等人，2024年b）。这些方法共同彰显了生成式人工智能支持个性化编程实践的潜力。

在我们之前的研究中（Na Nongkhai 等人，2023、2025），我们介绍了 ADVENTURE（基于多种编程语言本体的自适应学习支持系统），这是一个旨在帮助学习者进行多种语言编程练习的自适应学习支持系统。ADVENTURE 集成了 CONTINUOUS（Na Nongkhai 等人，2022），这是一种领域本体，用于定义多种编程语言（Python、Java 和 C#）共有的概念，以帮助学习者理解编程概念。此外，ADVENTURE 在教育环境中采用 Elo 评级系统（ERS）（Pel´anek，2016）作为自适应机制，为学习者推荐符合其技能水平的练习的练习。

在 ADVENTURE 中，学习者首先选择一种编程语言和一个概念，并完成前测以确定其初始难度级别（简单、标准或困难）；随后，系统会根据其技能提供相应的练习。系统会根据学习者的代码提交更新其技能水平，并推荐下一项练习。当学习者在困难级别达到阈值后，便可进入 CONTINUOUS 推荐的下一个概念，或自行选择下一个概念。在代码评估中，ADVENTURE 使用单元测试自动评估学习者代码，将代码运行结果与每道题定义的测试用例进行比较。因此，学习者针对每个测试用例只能获得正误信息，而没有解释性反馈。这一局限削弱了系统实现个性化学习的能力，因为系统既未说明解答错误的原因，也未指导学习者如何改进代码结构。

为解决这一局限，并进一步帮助学习者提升编程技能，本研究将 RAG 和生成式人工智能集成到 ADVENTURE 中，以增强系统功能，例如评估学习者代码、针对学习者的编程答案提供自动反馈，以及推荐下一项适合的练习。具体而言，通过在基于 RAG 的反馈框架中引入 GPT-4（系统开发时广为人知的大语言模型），ADVENTURE 能够在生成解释和建议之前，从 CONTINUOUS 本体和学习记录中检索相关信息。此项集成旨在帮助学习者理解针对其编程答案提供的反馈，并比较学习者对自适应方法和生成式人工智能所推荐的下一项练习的偏好。因此，本研究将探讨以下三个研究问题：

1. 如何将 RAG 有效集成到基于本体的自适应学习系统中，以支持编程教育中的形成性反馈？ 2. 在学习表现（包括学习成果和感知）方面，使用以下方法的学习者之间有何差异：(1) 不使用生成式人工智能的自适应方法；(2) 采用自适应方法，并使用生成式人工智能反馈（基于 GPT-4）和练习推荐；(3) 结合自适应推荐与生成式人工智能驱动的推荐，并提供生成式人工智能反馈的混合方法？ 3. 学习者如何看待生成式人工智能所提供反馈的帮助程度？

为回答研究问题，我们在 ADVENTURE 中开发了三种不同模式：自适应模式、生成式人工智能模式，以及整合自适应模式和生成式人工智能模式的模式。我们在一所泰国大学对 87 名本科生开展了实验，并收集了有关学习感知和学习日志的数据，随后对数据进行了分析。

我们的主要贡献是将知识图谱和用户交互历史与检索增强生成相结合，为大语言模型在教育领域的应用奠定了更强大、更可靠的基础。据我们所知，这是首批专门针对智能教育系统设计并评估此类集成框架的研究之一。作为对这一架构创新的补充，我们通过比较不同推荐策略和反馈机制下编程实践的真实学习表现，提供了新的数据驱动见解。这些实证结果不仅验证了我们的方法，也为未来开发个性化、基于大语言模型的教育方案提供了有价值的指导。

本文其余部分的组织结构如下：第2 节回顾了使用大语言模型生成编程代码自动反馈的相关工作，以及检索增强生成和大语言模型在教育系统中的集成。第3 节介绍 ADVENTURE 中三种不同模式的设计与开发。第4 节描述本研究采用的实验设置。第5 节报告结果，包括对问卷回答和学习日志的分析。最后，第6 节通过讨论主要发现、局限性和未来研究方向对本文作出总结。

2. 文献综述

2.1. 编程与教育情境中的大语言模型

生成式人工智能领域的大语言模型迅速发展，促使众多研究探索其在编程教育中设计和开发教学策略方面的应用。例如，Azaiz 等人（2024）研究了利用大语言模型为学生的编程练习生成自动反馈的应用。此外，多项研究探讨了 ChatGPT 和 Gemini 等人工智能聊天机器人的潜力，以制定能够有效支持学习者练习编程的教学策略（Kosar 等人，2024；George &Dewan，2024；Majumdar 等人，2024）。

此外，Zhang 和 Liu（2024）推出了 JavaLLM。这一模型基于 CodeQwen 1.5，使用文档、作业、社区讨论和代码反馈等 Java 专用资源进行微调。与基础模型相比，JavaLLM 在代码生成和问答方面表现更佳，并证明其在面向课堂的 Java 学习中效果显著。这凸显了领域专用大语言模型在提升教学质量和个性化学习方面的优势。然而，该模型的局限在于仅支持 Java 教育。Kazemitabaar 等人（2024）推出了 CodeAid，这是一款由 GPT-3.5 驱动、基于大语言模型的编程助手，旨在为编程教育提供教学支持，但不直接提供代码解答。CodeAid 在一门为期 12 周的大学 C 语言编程课程中部署，用于辅助调试、概念解释和伪代码支架式教学。学习者在课堂活动和家庭作业中使用 CodeAid，并报告称其回复准确率较高（79%），且感知到的实用性较强（86%）。教育工作者指出，CodeAid的可扩展性及其与学术诚信原则的一致性；学生则赞赏其私密且具备上下文感知能力的反馈。这项研究展示了在正式教育中以受控方式整合大语言模型，为编程学习提供合乎伦理且个性化支持的潜力。Tang 等人（2024）推出了 SPHERE，这是一个交互式系统，利用 GPT-4 支持教师为课堂编程练习和同伴讨论提供可扩展的个性化反馈。SPHERE 在一门大型 Python 课程中投入使用，生成包含问题、策略和示例的反馈草稿，供教师审核和定制。研究表明，GPT-4 能够生成高质量、结合上下文的反馈，且不会增加教师的工作量，这凸显了 SPHERE的提升个性化学习效果的潜力。

除了课堂教学系统之外，许多研究还将大语言模型融入在线个性化学习工具的开发中

教育领域。 Yang 等人（2024年）提出了 CREF，这是一种对话式修复框架，旨在利用大语言模型生成并迭代优化代码修正，为编程辅导教师提供支持。他们开发了 TutorCode，这是一个大型数据集，包含 1239 个错误的 C 语言编程样本++ ，并标注了辅导教师指导、解答说明和未通过的测试用例。研究人员利用该数据集对 12 个大语言模型进行了实验，发现提供教学语境的 GPT-4 达到了最高正确率（76.6%）。CREF' 的多阶段方法利用对话历史和上下文提示，提高了修复质量，同时将辅导教师的调试工作量减少了 71.2%，并将响应时间从 26.7 分钟缩短至 7.7 分钟。这些发现凸显了对话式大语言模型在编程教育中提供可扩展、高质量反馈的潜力。 Gabbay 和 Cohen（2024年）提出了一种在线编程慕课（大规模开放在线课程）教学方法，将自动化测试用例结果与 GPT生成的解释相结合，旨在提升学习者' 的理解能力和问题解决能力。该方法在一门大型 Python 入门课程中实施，并通过分析学习者交互和表现进行评估。研究结果表明，参与组合反馈的学习者修正率有所提高，在调试代码时也更具持久性。此外，学习者表示，他们认为反馈很有用且清晰。本研究凸显了将大语言模型生成的反馈与自动化评估相结合，在编程教育中支持可扩展个性化学习的有效性。类似地，Li 等人（2024b）推出了 ProgMate，这是一种智能编程辅助系统，将大语言模型与知识图谱推理相结合，为大规模编程课程中的个性化学习提供支持。系统通过大语言模型引导的抽象语法树（AST）表示分析错误代码，识别学习者困难，并将其映射到知识图谱中的相关概念。ProgMate 采用概率技能建模评估熟练度并生成自适应反馈，根据学习者的理解情况以及 C语言编程的概念结构，引导其沿个性化路径学习。研究报告显示，该系统具有较高的预测准确率，学习者反馈也很积极，尤其是在采用 GPT-4 时。这些结果展现了将大语言模型与结构化教育框架相结合，以提升编程教学可扩展性的前景。

如上所述，多项研究将大语言模型集成到系统中，以提供形成性反馈，旨在减轻教师的 工作负担，例如 SPHERE（Tang 等人，2024年）和 CREF（Yang 等人，2024年）。这些系统将大语言模型用作助手，根据学习者的 代码生成反馈，并提供建议供教师审核后再发送给学生。与之相反，CodeAid（Kazemitabaar 等人，2024年）和 Gabbay 和 Cohen（2024年）等其他研究则侧重于借助生成式人工智能为学习者提供促进学习的反馈，同时避免直接给出代码解答。因此，本研究旨在通过利用大语言模型自动生成反馈并为学习者推荐下一项编程练习，增强 ADVENTURE。ProgMate（Li 等人，2024b）提供个性化学习路径和个性化反馈，而 ADVENTURE 则整合了多种编程语言的本体（称为 CONTINUOUS），并在教育环境中采用基于 ERS 的自适应机制。该机制与生成式人工智能模型（GPT-4）生成的大语言模型反馈协同运行。

2.2. 面向大语言模型的检索增强生成技术与知识图谱

虽然以往研究主要关注大语言模型如何支持编程学习，但很少有研究探讨实现个性化反馈的底层技术机制。如第2.1节所示，大语言模型凭借理解语义上下文并生成高质量文本回复的能力（Gan 等人，2023），在编程教育中展现出巨大潜力，使其能够充当智能虚拟导师，提供即时且个性化的反馈。此外，近期研究表明，大语言模型具备强大的跨学科推理能力，这对于应对复杂的教育场景很有价值（Xu 等人，2024a）。然而，大语言模型在教育领域的应用面临重大挑战。大语言模型容易产生幻觉，即给出看似合理但实际上不正确的答案，这可能会误导学习者（Li 等人，2024a）。提升事实可靠性和适应性仍是教育人工智能研究的重要方向。

为缓解这些局限，检索增强生成（RAG）应运而生：大语言模型'的内部知识与外部数据源相结合，使其能够在生成响应的过程中检索相关且最新的信息（Lewis 等，2020）。该架构通常包括一个检索器，用于定位最相关的文档；以及一个生成器，用于将这些文档与查询结合，生成具有背景信息的响应。通过以检索到的证据为基础进行生成，RAG 可减少幻觉并提高事实准确性（Lewis 等，2020）。在教育领域，Dong 等（2023）使用 LangChain（Chase，2022）开发了一个人工智能导师，将课程材料整理到知识库中，并使用 RAG 检索与学生查询相关的信息。检索到的数据会与问题结合，形成新的提示，然后由大语言模型处理以生成响应。

除了基于检索的方法外，知识图谱（KGs）也日益被用于构建教育知识并促进个性化学习。早期研究利用以课程为中心的本体，将知识点与现有语言学习材料联系起来（Wang 等人，2014），而较新的研究则提出了本体驱动的强化学习框架，可根据学生表现调整教学策略，并提供个性化学习建议（Hare & Tang，2024年）。通过明确建模概念结构，知识图谱能够为学习者提供更易于理解、更有针对性的反馈。

将知识图谱与检索增强生成结合，能够进一步提升推理和检索能力。传统检索增强生成难以处理复杂的多跳推理和非结构化数据，导致其在较长或概念密集型查询中的检索准确性下降（Chang & Zhang，2024年；Xu 等人，2024年b）。知识图谱增强的检索增强生成通过提供结构化的关系上下文来引导检索过程并增强逻辑一致性，从而应对这些挑战。近期方法通过纳入基于社区的图结构或经解析的历史数据，来提高检索精确率并进行事实核查（Chang & Zhang，2024年；Xu 等人，2024年b）。这种整合在教育情境中尤为相关，因为学习材料往往构成相互关联的知识网络。

总而言之，大语言模型、检索增强生成和知识图谱的融合，为构建智能教育系统奠定了坚实基础。检索增强生成提高了事实准确性，知识图谱提供了结构和可解释性，而二者的结合则能够提供自适应、感知背景信息的学习支持。在这些进展的基础上，ADVENTURE 将知识图谱融入检索增强生成流程，以查找并推荐与学习者当前遇到困难的内容密切相关的练习。系统利用学习者的会话历史，在其练习过程中提供推荐；同时也会收集详细的日志数据，以便后续分析学习行为。此外，我们还探讨了三种不同策略下学习表现的差异，其中两种策略进一步与 GPT-4 集成。知识图谱驱动的检索增强生成、感知会话信息的推荐以及自适应机制共同构成了 ADVENTURE的设计特色，为个性化编程实践提供了一种新颖的方法。ADVENTURE 中实现的各项策略详见第 3 节。

3. ADVENTURE 中三种模式的设计与开发

本研究基于此前开发的 ADVENTURE 系统（Na Nongkhai 等人，2023，2025），该系统提供编程

练习，并根据学习者的技能量身定制。为回答研究问题，我们通过实现三种不同模式对 ADVENTURE 进行了扩展：

（1）自适应模式，名为 ADVENTURE-2.0，使用单元测试评估学习者的代码：将程序的输出与预期结果进行比较，并根据测试结果提供反馈（按单元测试提供）。此外，学习者的技能会通过 Elo 评级系统（ERS）进行更新（Pel´anek，2016），使系统能够根据每位学习者的当前技能推荐练习。

（2）两种采用不同推荐策略的生成式人工智能模式，均使用 GPT-4 模型和 CONTINUOUS 知识图谱来评估学习者代码，并根据其提交内容生成形成性反馈。（a）生成式人工智能模式名为 ADVENTURE-3.0，使用 GPT-4 根据学习者的表现推荐下一项合适的编程练习。只有当推荐的练习此前已成功完成时，才会提供自适应模式推荐的备选练习。（b）生成式人工智能-自适应混合模式名为 ADVENTURE-3.1，为学习者提供在生成式人工智能推荐和自适应模式推荐之间进行选择的选项。在此模式下，学习者可以拒绝 GPT-4 的推荐，转而选择自适应模式推荐的练习。不过，如果学习者选择了生成式人工智能模型的推荐，而推荐的练习恰好是重复的，系统会提醒学习者，并提供更改决定、采用自适应模式推荐的选项。

后两种模式的生成式人工智能流程会在学习者提交代码后启动，如图1所示。系统会将提交信息，包括个性化标识符、编程问题、编程语言、学习者'的代码，以及用户的聊天历史，从前端发送到后端，作为生成式人工智能模型的提示数据。为满足检索增强生成流程的特定要求，设计了两个提示，如图A1所示。第一个提示是复合提示，用于验证所提交代码的正确性、识别错误、提供修正后的解决方案，并解释相关错误观念。它还会根据学习者过去的表现推荐新的练习，以弥补用户技能中发现的特定知识缺口。第二个提示，即问题改写提示，会将可能含糊不清的查询转换为清晰、独立的问题，确保检索和生成式响应在上下文中准确无误。

提示数据还包含一个知识图谱，作为系统检索增强生成流程的基础。该图谱基于多种编程语言的 CONTINUOUS 本体构建，将编程问题表示为节点，并捕捉问题之间的关系。借助该图谱，系统能够识别与学习者当前问题密切相关的练习节点。这些节点存储在向量数据库中，以支持高效的语义搜索，并检索最适合学习者当前进度的练习。图谱可使用 Cypher 图查询语言进行更新，以查询、修改和扩展数据库中的节点。

使用 LangChain 框架构建的系统（Chase，2022），随后将学习者'的查询分解为多个组成部分，并通过历史感知检索器整合学习者'的输入及其交互历史。该检索器会访问向量存储库，根据学习者'以往的提交和表现，选取与背景信息相关的练习和信息。在验证所提交代码的正确性后，系统将检索到的背景信息提供给 LangChain 问答链，由其生成形成性反馈，包括在出现错误时对错误进行解释。然后，系统利用知识图谱和学习者'的历史记录，推荐后续练习，以针对特定知识缺口，同时避免不必要的重复。

整个交互历史都会存储在记忆组件中，使系统能够跨多个会话保留背景信息，并逐步提供个性化支持。这种将知识图谱驱动的检索与记忆增强反馈相结合的方法，为提供适应性强、能够感知背景信息且依据每位学习者不断变化的需求量身定制的编程实践，提供了一种新颖而有效的途径。

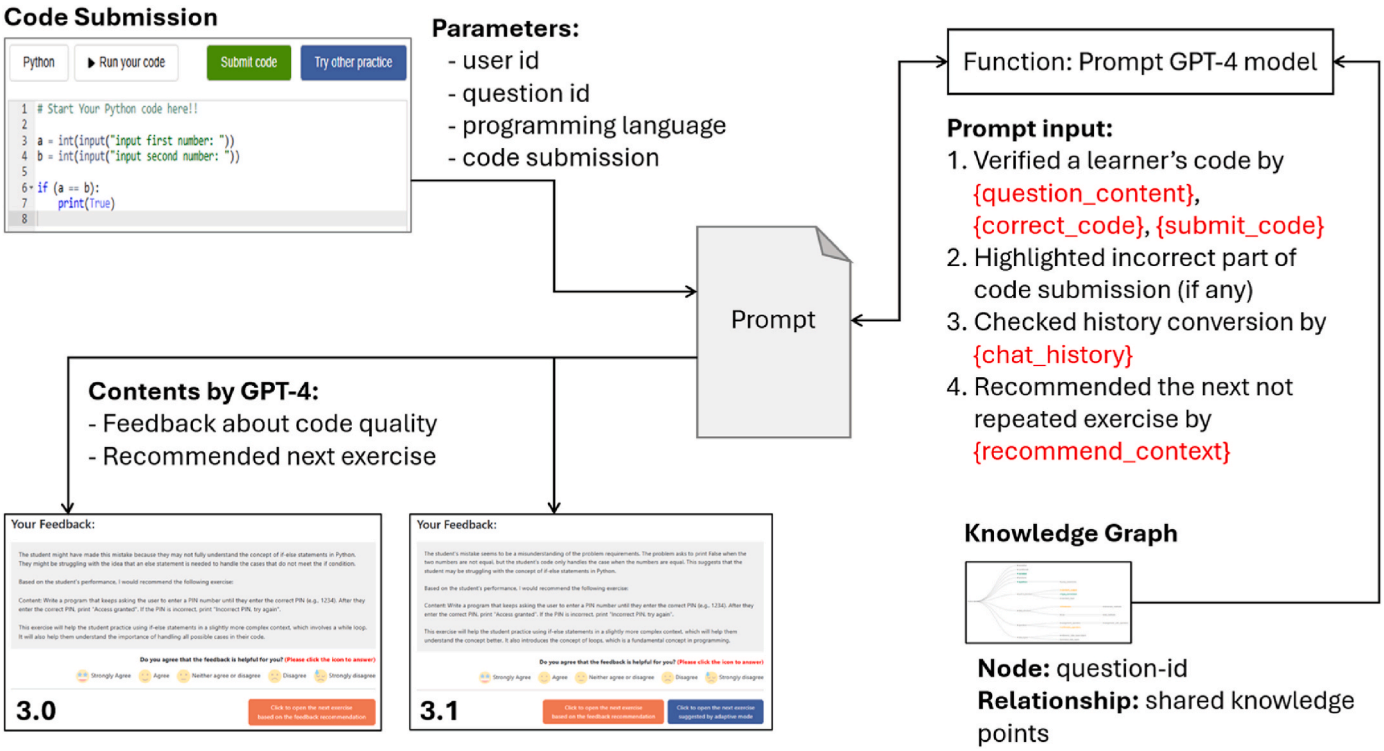

图1. 生成式人工智能在 ADVENTURE 中的集成。

3.1. 自适应模式（*ADVENTURE-2.0*）

自适应模式（ADVENTURE-2.0）在 ADVENTURE-1.1 的基础上整合了多项增强功能（Na Nongkhai 等人，2025）。例如，更新后的系统新增了以下功能：（1）进度条，用于跟踪学习者在各个难度级别上的技能（如图 2的高亮框所示）；（2）扩展提示，为每道题目提供额外的知识点（如“本题提示：” 下方的图 2 和图 3所示）；以及（3）仅向学习者显示错误的测试用例。

在 ADVENTURE 所有模式的初始流程中，学习者登录后会选择一种编程语言和一个概念进行练习。如果所选概念此前尚未尝试过，系统会提示学习者完成前测，以确定其初始难度级别。根据前测结果，学习者将开始完成分配给相应级别的练习。这些步骤是 ADVENTURE 所实现的所有模式的入口。

此模式的具体流程包含四个步骤，用于确定下一个合适的练习。学习者向系统提交代码后，系统会启动以下步骤：（1）使用单元测试评估提交的代码；（2）更新学习者技能以及已尝试练习的难度；（3）确定一个难度级别与学习者当前技能最为匹配的练习；（4）向学习者推荐所选练习。通过这一循环，每次提交后都会持续更新学习者技能和练习难度级别，确保后续练习始终与学习者当前技能相匹配。

为了评估学习者的代码在此模式下的表现，我们采用单元测试方法来验证每道编程问题中的测试用例。 图 3展示了一道编程任务示例。如果学习者提交了错误的代码，例如条件写错且遗漏了 else 分支（如图 4所示），因此未能通过当前问题的任何测试用例，系统便会向学习者显示这些未通过的测试用例，如图 5所示。

相反，如果代码成功通过所有测试用例，系统会提示学习者继续完成下一道练图 6所示。在此模式下，学习者会收到系统自适应机制推荐的练习。对于代码提交反馈，系统仅显示未通过的测试用例，并提供有限的解释。为解决这一局限，我们设计并实现了由生成式人工智能生成的形成性反馈。

3.2. 生成式人工智能模式（*ADVENTURE-3.0*）

生成式人工智能模式（ADVENTURE-3.0）是在 ADVENTURE-2.0 的基础上，通过将 GPT-4 模型（图1）集成到系统中开发而成的。在此模式下，开发了一个应用程序接口，用于连接使用 React（https://reactjs.org/，访问日期：2022年10月1日）开发的前端与使用 Python 构建的后端，以调用 GPT-4 模型（OpenAI，2023）。收到 GPT-4 模型的响应后，前端会在反馈界面上显示两项内容：（1）关于代码质量的反馈；（2）推荐的练习，如图7左侧所示。

在代码提交的反馈界面中，生成式人工智能模型生成的代码反馈以方框形式标注于图7中的第一个框内，而推荐的下一道练习的内容则以方框形式标注于图7中的第二个框内。继续之前，系统还要求学习者确认是否同意该反馈。与 ADVENTURE-2.0 中呈现的反馈（图5）相比，ADVENTURE-3.0 中的反馈提供了更详细的解释，帮助学习者识别并理解自己觉得有难度的概念。此外，它还针对下一道练习的推荐提供了具体详情。

不过，该模式下生成式人工智能模型提供的练习推荐仍有一些局限。尽管提示中要求根据聊天历史避免推荐重复问题，GPT-4 仍有时会推荐学习者已经完成的问题。在这种情况下，弹出对话框会询问学习者是否希望重复进行推荐的练习。如果学习者选择重复练习，系统会将该练习重新分配给学习者；否则，系统会通过自适应机制提供一道新练习。自适应流程是生成式人工智能模式中的一种替代路径，仅在这一特定情形下（即系统推荐了重复练习时）才会启用。

3.3. 自适应模式与生成式人工智能模式的集成（*ADVENTURE-3.1*）

生成式人工智能-自适应混合模式（ADVENTURE-3.1）在 ADVENTURE-3.0 的基础上开发，结合了自适应推荐和生成式人工智能推荐。如图7所示，与3.0相比，主要区别在于，在3.1中，自适应模型始终作为一个选项提供，而不只是当 GPT-4 推荐已完成过的练习时才可用。在此模式下，学习者可以选择接下来进行由生成式人工智能或自适应方法推荐的编程练习（点击蓝色按钮即可）。

换言之，ADVENTURE-3.1引入了一种推荐流程：学习者首先会看到生成式人工智能模型推荐的练习。如果他们不认同生成式人工智能推荐，就可以像在 ADVENTURE 2.0 中一样，使用自适应模式推荐。只有在拒绝生成式人工智能推荐后，学习者才会看到自适应推荐。如果学习者也拒绝自适应模式的推荐，系统将提供由自适应模式推荐的另一项练习。

如果他们最初接受生成式人工智能推荐，则会像在 ADVENTURE 3.0 中一样分配练习。在这种情况下，只有当学习者已经完成该练习时，系统才会提醒他们是否希望通过自适应模式获取另一项推荐；否则，系统将向他们提供生成式人工智能推荐。ADVENTURE-3.1 将自适应建议和生成式人工智能建议整合到推荐

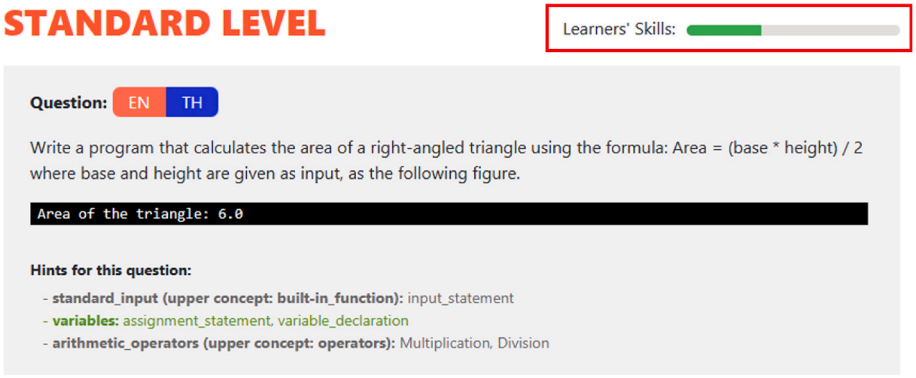

图 2. 各难度级别下学习者技能进度条的显示（红框中突出显示）。

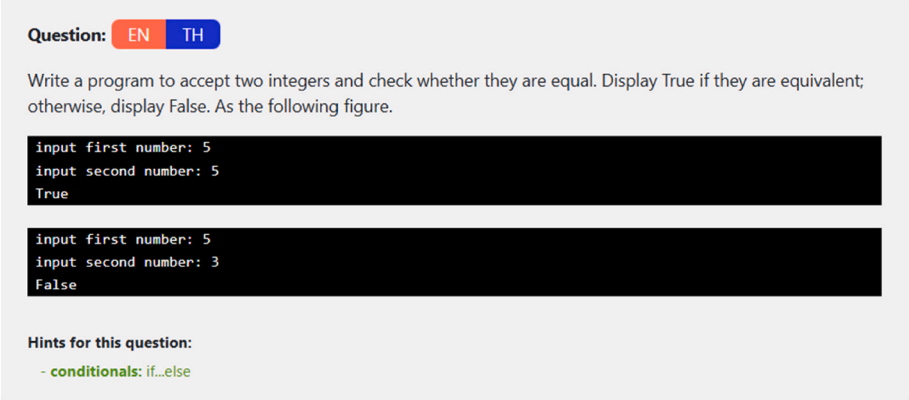

图 3. 编程问题示例。

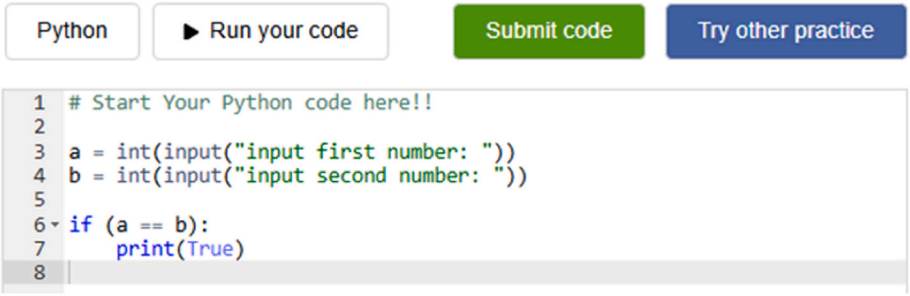

图 4. 学习者代码示例。

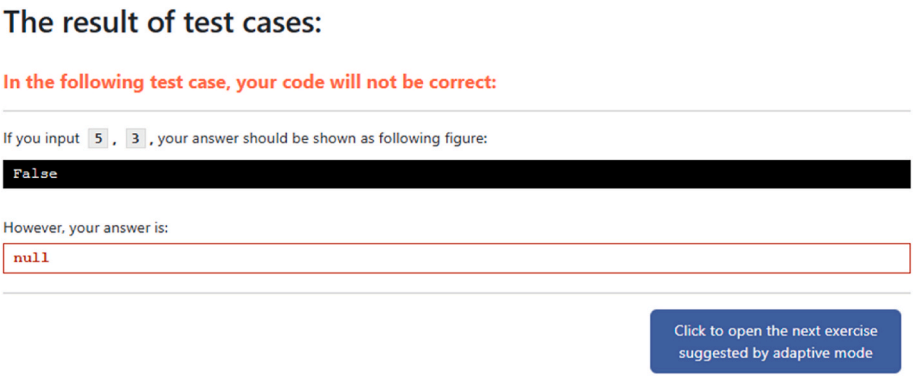

图 5. 未通过的测试用例示例。

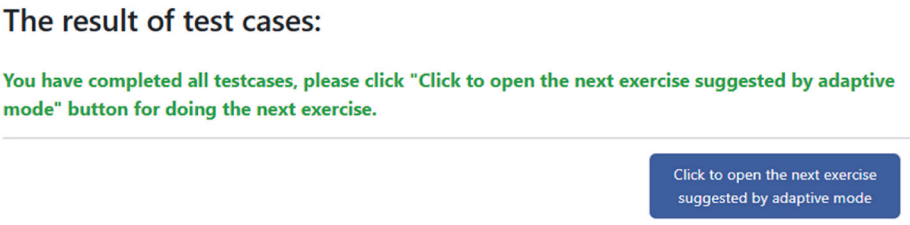

图 6. 通过所有测试用例后的消息。

策略中，为学习者提供了一种更加灵活的方式，使他们可以在两种推荐之间进行选择。

4. 实验描述

4.1. 参与者与测量技术

来自泰国一所大学、主修信息技术的一年级和二年级本科生共87人参与了此次

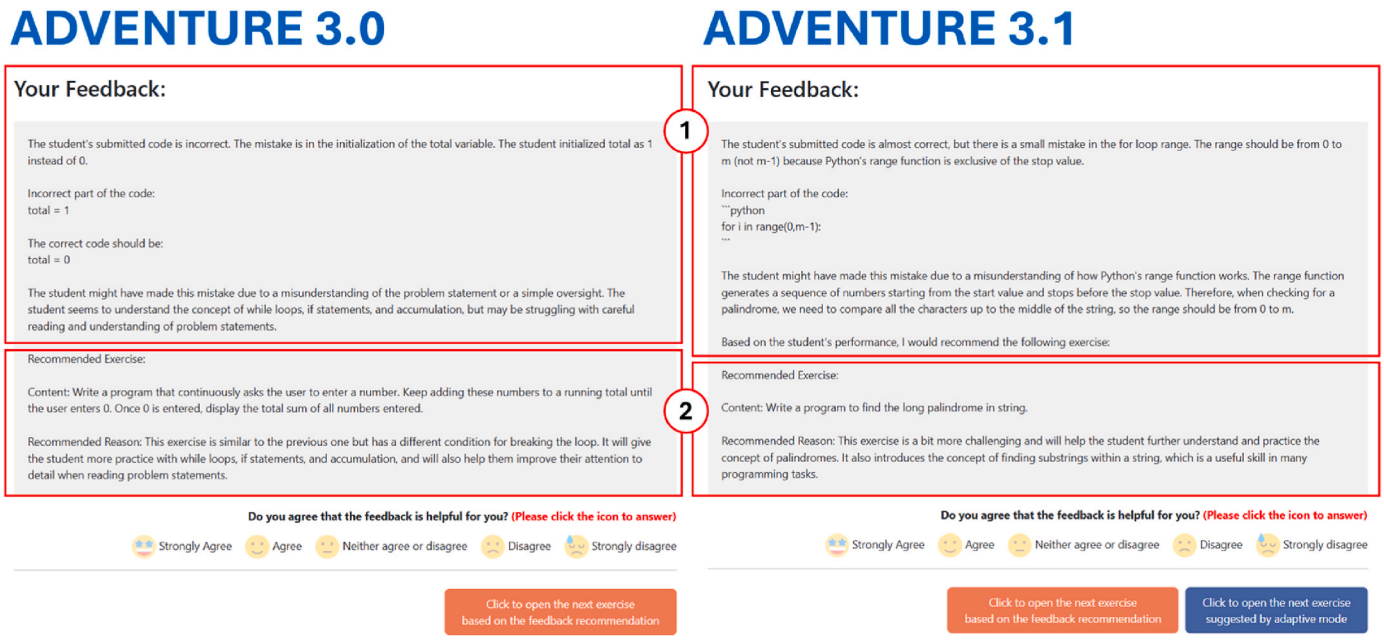

图7. ADVENTURE 3.0 和 3 中代码提交反馈及下一项练习推荐示例 .1.

实验。 一年级学生此前修读过一门编程课程，包含30小时讲座和45小时实践活动。二年级学生已修完两门编程课程，每门均包含30小时讲座和45小时实践活动。为确保公平，参与者被分配到三个实验组，每组由15名一年级和14名二年级本科生组成，共29名参与者。分组如下：(1) A组使用自适应模式（ADVENTURE-2.0），由23名男性和6名女性组成；(2) B组使用生成式人工智能模式（ADVENTURE-3.0），由22名男性和7名女性组成；(3) C组使用生成式人工智能自适应模式（ADVENTURE-3.1），由28名男性和1名女性组成。由于本研究的对象是编程经验有限的初学者，因此难以在各组之间实现性别代表性的均衡。此局限将在第6节讨论。

本研究采用了两种测量技术：(1) 学习日志，包括每次编程练习的学习成果、学习者对代码提交后所收到的每条反馈的感知（仅针对B组和C组），以及其他学习行为；(2) 学习感知问卷。该问卷以泰语编制，依据（Wang 等人，2014，2020，2024年），并在经验丰富的教师指导下进一步完善，用于评估学习者对其的学习体验的感知。问卷包含26个条目，采用七点李克特量表测量，其中1–3分表示不同程度的不同意，4分表示中立，5至7分表示不同程度的同意。条目分为四个维度：(1) 心理努力，反映理解实验目的并完成学习活动所付出的努力；(2) 心理负荷，反映分心和压力；(3) 技术接受度，包括易用性和感知有用性；(4) 满意度。所有参与者均在学习活动结束后完成问卷。

4.2. 实验程序

图8 展示了本研究采用的实验程序。所有实验组均有15天时间以指定模式使用 ADVENTURE系统。选择一个编程概念后，系统会向参与者提供一份由三个问题组成的前测，每个问题分别代表一个难度级别）。如果参与者未能答对某个问题，后续编程练习将被分配到该问题对应的难度级别。

除了方便参与者随时访问系统外，我们还在实验期间的三个不同日期（第1天、第7天和第15天）开展了现场课程。每次课程

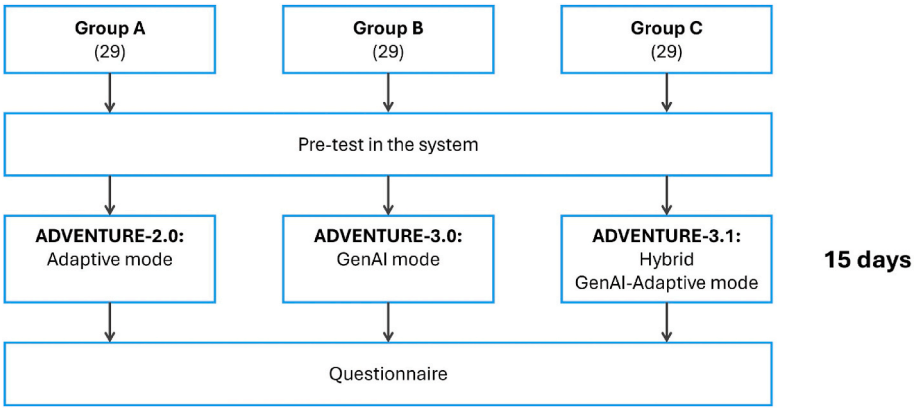

图8. 实验程序。

持续1.5小时，旨在让参与者在可控环境中使用系统时，监测他们的学习进度。第1天，我们为每位参与者提供了预先分配的系统账户，并指导他们完成初始登录流程和系统使用。第7天，我们再次监测参与者的活动。这一次，我们鼓励他们在使用系统时遇到漏洞或技术困难时寻求帮助。第15天，我们请参与者填写问卷，以反馈他们的学习体验。实验期间的其他课程均未在课堂上进行监测，并鼓励参与者在课外空闲时间使用我们的基于网络的系统。

5. 结果

5.1. 问卷回答分析

表1 总结了各组87名参与者对学习感知问卷回答的分析结果。每位参与者均需完成26道题，这些题目分为四类：心理努力（ 2 α = 0 582 4 3 道题，.）；心理负荷（道题，包括涉及三项系统功能导致的分心的题目，以及一道测量压力的题目，$\alpha = 0.685$）；技术接受度（10道题，涵盖对五项系统功能的易用性和有用性的评价，$\alpha = 0.853$）；以及满意度（10道题，$\alpha = 0.904$）。此外，Harman'单因子检验显示，第一因子解释了总方差的37.70%，低于50%。因此，本研究未将共同方法偏差视为严重问题。

对于"心理努力"，各实验组在理解使用该系统的目的方面的平均评分接近中立点，为4.62。使用生成式人工智能模式的组，其标准差略高。这表明，所有参与者对理解使用 ADVENTURE 的目的所需的心理努力均持相近的中立评价，且这种感知在不同模式间保持一致。同时，各组在学习活动项目上的平均评分均高于中立点：A组为5.48（标准差 = 1.06），B组为6.24（标准差 = 1.27），C组为5.52（标准差 = 1.21）。这些结果表明，与其他模式的参与者相比，生成式人工智能模式（B组）的参与者在练习编程活动时付出了更多努力。此外，方差分析结果显示，各组间存在显著差异（$F(2,84) = 3.795$，$p < 0.05$, $\eta^2 = 0.083$）。事后两两比较表明，B组在学习活动中付出的心理努力显著高于A组（ΔM =0.759，$p = 0.017$）和C组（ΔM = 0.724，$p = 0.022$）。

关于"心理负荷"，A组和B组的平均分心评分略低于中立点，分别为3.86 （标准差 = 1.52 ）和 3.75 （标准差 = 1.47 ）。相比之下，C 组的评分为 4.46（标准差 = 1.39），略高于中立点，这表明C组参与者在使用系统时感受到的分心程度略高。ADVENTURE-3.1 要求学习者在生成式人工智能模式和自适应模式之间选择下一项练习，这可能增加了复杂性，并导致C组的分心评分略高。与此同时，B组和C组的平均压力评分分别为 5.55（标准差 = 1.57）和 5.17（标准差 = 1.61），而A组报告的压力略低（均值 = 4.90，标准差 =1.84）。这表明，使用集成生成式人工智能的系统的参与者在练习编程时可能遇到了一些挑战，可能是因为练习由生成式人工智能模型推荐，从而导致感知压力更高。然而，方差分析结果表明，各组之间的差异均无统计学意义。

关于"技术接受度，" 五项系统功能（包括编程概念的图形可视化、编程练习网页、每道编程问题中的提示、针对代码提交的反馈，以及下一步编程练习的建议）的易用性平均评分在三个组中均超过 5 分。（最后两项功能的评分将在第 5.3 节中作进一步讨论。）同样，所有组的有用性平均评分也都超过 5 分。这些结果表明，无论使用哪种模式，该系统都易于使用，并且有助于练习编程。此外，方差分析结果表明，各组之间没有显著差异。与此结果一致，所有组的总体满意度平均评分均超过 5 分。此外，方差分析结果表明，各组之间没有显著差异。这些结果表明，所有组的参与者总体上都认为 ADVENTURE 适合用于编程练习，并对此感到满意。

5.2. 学习日志分析

在实验期间，学习者的ADVEN-TURE 活动会自动记录为学习日志。这些日志包括首次登录时填写的编程体验问卷答复、各编程概念的前测结果、学习者的技能以及代码提交。对一年级和二年级学生基线编程熟练度的事后分析表明，他们先前编程知识的差异很小：一年级学生的前测平均分为 51.35%，二年级学生为 53.53%。参与者共提交了 4956 份代码：A组 1633 份、B组 2030 份、C组 1293 份。为分析学习成果，我们将代码提交分为四个不同特征：（1）正确答案提交；（2）错误答案提交；（3）缺少必要编程逻辑的提交；（4）请求另一道练习的频率。图9汇总了三个实验组学习日志中这四项特征。

对于"正确答案提交，" 如图9中的（1）所示，条形图表明，C组的正确提交平均数量最高。此外，方差分析结果表明，各组之间均存在显著差异（F (2,84 = 6 <) .343, 0 01 η^2 = 0 131 p .,. ）。事后比较显示，A组的正确提交数量显著少于B组（Δ均值 = 0.091, p <

表1学习感知问卷回答摘要。

		心理努力		心理负荷		技术接受度		满意度
		目的	学习活动	分心	压力	易用性	有用性	
A组（ADVENTURE 2.0）	Mean	4.62	5.48	3.86	4.90	5.59	5.72	5.34
	S.D.	1.63	1.06	1.52	1.84	0.72	0.72	0.75
B组（ADVENTURE 3.0）	Mean	4.62	6.24	3.75	5.55	5.57	5.66	5.37
	S.D.	1.90	1.27	1.47	1.57	0.85	0.89	1.02
C组（ADVENTURE 3.1）	Mean	4.62	5.52	4.46	5.17	5.63	5.77	5.38
	S.D.	1.63	1.21	1.39	1.61	0.97	0.82	1.02
单因素方差分析	F	0	3.795	1.994	1.117	0.042	0.136	0.016
	P	1	0.026*	0.142	0.332	0.959	0.873	0.984

注：方差分析中的 df(2, 84)，$^{**}p < 0.01$，$^{*}p < 0.05$。

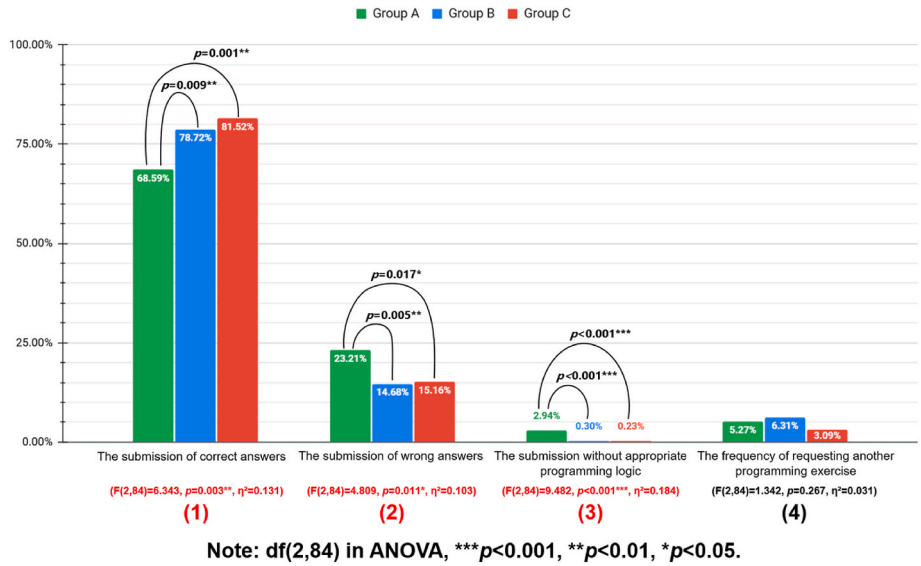

图9. ADVENTURE中三种模式下四项学习日志特征的比较：(1) 正确答案提交；(2) 错误答案提交；(3) 提交内容缺少恰当的编程逻辑；(4) 请求另一道编程练习的频率。

0.01），且C组（Δ均值 = 0.114，p < 0.01）。然而，生成式人工智能模式与混合模式之间没有显著差异，这表明两种模式在正确代码提交率方面均优于自适应模式；混合模式的均值差异略高于生成式人工智能模式，说明与其他模式相比，混合模式可能为编程实践带来更显著的益处。

对于“提交错误答案”，如图9中的（2）所示，条形图显示，与其他组相比，A组提交的错误答案数量最多。方差分析结果表明，各组之间存在显著差异（F (2,84) = 4.809，p < 0.05，$\eta^2 = 0.103$）；两两比较显示，A组的错误提交显著多于B组（ΔM = 0.070，p < 0.01）和C组（ΔM = 0>.059，p < 0.05）。这一发现表明，与单独使用自适应模式相比，基于生成式人工智能的反馈和练习推荐可能帮助参与者更有效地发现并纠正错误。然而，B组和C组之间没有发现显著差异，这可能是因为两组使用的系统都融入了生成式人工智能。

对于“提交缺少恰当编程逻辑的代码”，这一情况指参与者提交空代码，或提交的代码缺少与任务目标相符所需的必要逻辑，无论输出看起来是否正确。此类提交不同于错误代码提交，后者通常是尝试解决问题，但结果不正确。如图9中的（3）所示，条形图显示，与其他组相比，A组提交的缺少必要编程逻辑的代码数量更多。方差分析结果表明，各组之间存在显著差异（F (2,84) = 9.482，p < 0.001，$\eta^2 = 0.184$）。这些发现表明，与使用另外两个融入生成式人工智能模型的版本的参与者相比，使用自适应模式与系统互动的参与者更有可能提交缺少必要编程逻辑的代码。A组与B组（ΔM = 0.040，p < 0.001）以及A组与C组（ΔM = 0>.041，p < 0.001）之间的两两比较均显示存在显著差异，这也支持了这一发现。

对于“请求另一道编程练习的频率，”即参与者放弃当前练习并选择尝试其他练习的情况，方差分析结果表明，各组之间没有显著差异。然而，如图9（4）所示，C组平均请求替代练习的频率低于其他组的参与者。这表明，使用自适应模式与生成式人工智能模式集成的参与者，可能比仅使用生成式人工智能模式或自适应模式的参与者获得了更合适的练习推荐。这是因为参与者可以选择由生成式人工智能模式或自适应模式推荐的下一道练习。

总而言之，整体结果（图9）表明，与其他组相比，C组参与者提交的正确答案数量更多，代码提交中缺少必要编程逻辑的情况更少，且请求替代练习的次数也更少。这表明，针对代码提交提供更全面的反馈，并允许学习者选择由生成式人工智能或自适应算法推荐的下一项编程练习，可能有助于提升编程练习活动的效率。

5.3. 学习者对生成式人工智能所生成反馈的回答分析

为回答本研究的第三个研究问题，我们收集并分析了学习者对以下问题的回答：“您是否同意这些反馈对您有帮助？”，如图7底部所示。我们共收集到来自B组和C组参与者的3154份回答。

图10中的条形图展示了两个实验组对生成式人工智能模型所生成反馈的认同程度，采用五点李克特量表评分，范围从非常同意（5）到非常不同意（1）。B组和C组的平均评分为4.01（标准差= 1.27），表明大多数参与者认为GPT-4生成的反馈很有帮助。此外，*t*检验表明，B组（均值= 4.27，标准差=0.91）与C组（均值= 4.15，标准差= 0.87，t（56）= 0.831，p =0.409）之间没有显著差异。

表2 展示了学习者对两项系统功能——代码提交反馈和下一道编程练习推荐——在技术接受度维度上的看法。对于“代码提交反馈，”所有组在易用性和有用性方面的平均评分均高于5，表明参与者略微认同这项功能易于使用且有益。此外，如表3所示，定性反馈显示，虽然参与者认为 ADVENTURE-2.0 中由单元测试生成的简单反馈易于理解且有帮助（AP-1、AP-2），但有些

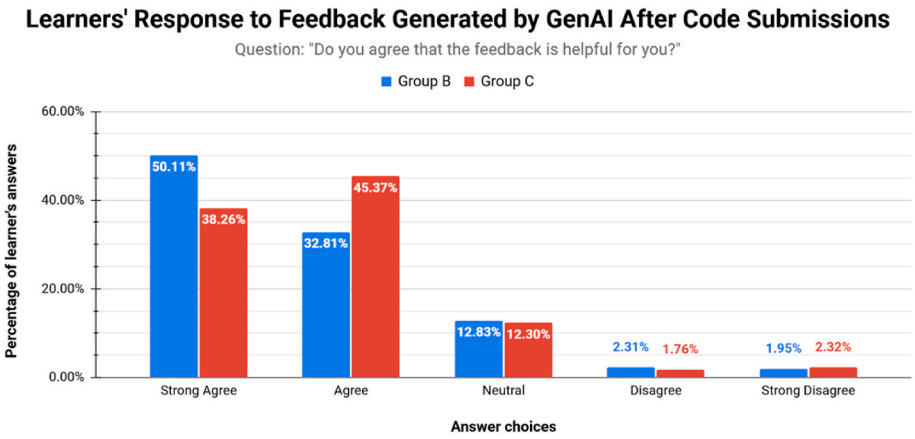

图10. 学习者对 ADVENTURE-3.0 和 ADVENTURE-3.1 中生成式人工智能所生成反馈的回答。

表2“代码提交反馈”和“下一道编程练习推荐”功能的技术接受度。

		代码提交反馈		下一道编程练习练习	
		易用性use	有用性	易用性use	有用性
A组（ADVENTURE 2.0）	Mean	5.38	5.76	5.48	5.45
	S.D.	1.18	0.95	1.24	1.12
B组（ADVENTURE 3.0）	Mean	5.14	5.69	5.59	5.48
	S.D.	1.25	1.26	1.35	1.35
C组（ADVENTURE 3.1）	Mean	5.38	5.76	5.41	5.55
	S.D.	1.59	1.24	1.50	1.43

参与者希望对自己的编码错误获得更详细的解释，并希望系统能更准确地评估他们的答案（AN-1、AN-2）。相比之下，B组和C组的几位参与者对生成式人工智能生成的反馈给予了正面评论，认为这些反馈很有帮助（BP-1至3、CP-1至5）。这些发现表明，尽管所有组的平均评分没有显著差异，但定性反馈凸显了生成式人工智能生成的解释可能更有助于学习者理解自己的代码。

对于“下一道编程练习推荐”功能的易用性和有用性，三个组的平均评分均高于 5，表明参与者略微认同该功能易于使用且有益。然而，根据学习日志数据，尽管提示 GPT-4 不要推荐重复的练习，B组仍出现了较高比例的重复练习推荐（76.95%）。此外，近半数学习者（31.69%）拒绝重复已经成功完成的练习。本研究将重复题目定义为学习者此前已正确解答、但随后又被系统再次推荐的练习。相比之下，C组中有 71.69% 的学习者选择不采纳生成式人工智能的推荐，而直接选择自适应建议。与 B组类似，C组也出现了生成式人工智能重复推荐练习的较高比例（71.54%）。这些定量发现与定性反馈一致：B组（BN-1、BN-2）和 C组（CN-1、CN-2）的参与者都指出练习经常重复。

此外，学习者的 定性意见经过分析，并分为三类反馈（错误解释、代码建议和概念澄清），如表4所示。大多数意见被归类为错误解释，这表明学习者重视能够指出并解释错误的反馈，因为这有助于他们认识到代码出了什么问题。还有少数意见被归类为代码建议，这类反馈提供提示或指导，帮助改进代码逻辑或得出正确的解决方案。此外，还有一些意见被归类为概念澄清，这表明学习者也认可有助于他们在纠正错误之外理解编程逻辑的反馈。这说明，反馈可能有助于学习者理解编码错误、正确的解决方案以及逻辑。

6. 结论与讨论

6.1. 讨论与启示

近年来，研究者越来越多地探索利用大语言模型生成自动化编程反馈。例如，Tang 等人（2024年）推出了 SPHERE，该系统整合 GPT-4，在将反馈草稿提供给学习者之前，先呈交指导教师审阅。同样，Gabbay 和 Cohen（2024年）在 Python 大规模开放在线课程中实施了 GPT 生成的解释性反馈，而 Li 等人（2024b）提出了 ProgMate，该系统利用 GPT-4 为 C语言编程提供形成性反馈。这些研究凸显了基于 GPT 的模型在提供准确且有助益的反馈方面的有效性，有助于提升学习者的 编程实践。

基于这些优势，本研究在检索增强生成框架下将 GPT-4 集成到 ADVENTURE 中，以回答第一个研究问题。检索增强生成设计使 GPT-4 能够在生成解释、建议和下一项练习推荐之前，从 ADVENTURE 的本体和学习者历史数据中检索相关的概念与练习信息。这种方法降低了生成泛泛而谈或忽视背景信息的回复的可能性，使反馈能够更贴合每位学习者的学习进展和过往活动。

为评估这种集成的有效性并回答第二个研究问题，本研究考察了三种方法下的学习感知和学习日志：（1）不使用生成式人工智能的自适应方法；（2）采用生成式人工智能反馈（使用 GPT-4）和练习推荐的自适应方法；（3）结合自适应推荐与生成式人工智能驱动的推荐，并采用生成式人工智能反馈的混合方法。

在学习感知方面，结果表明，参与生成式人工智能模式（B组）的参与者在编程练习中付出的努力显著多于其他组。此外，B组参与者还报告称，在练习题中感受到更大的压力，这表明生成式人工智能模型推荐的练习可能对学习者更具挑战性。相比之下，参与自适应模式（A组）的参与者报告的努力和压力水平均最低，这表明自适应机制可能会为学习者分配较少具有挑战性的练习。

对学习日志的分析（图9）显示出显著的

表3学习者对 ADVENTURE 的评论。

	正面评论	负面评论
A组 (ADVENTURE 2.0)	AP-1：“输出很容易理解。” AP-2：“指出测试用例出错位置的函数很有帮助。” AP-3：“练习种类丰富。”	AN-1：“它只提供输出；我希望能有更多解释。” AN-2：“答案检查需要改进。”
B组 (ADVENTURE 3.0)	BP-1：“反馈很有帮助。” BP-2：“我喜欢反馈中的指导功能。” BP-3：“代码后的反馈提供了出色的指导和知识。” BP-4：“就……而言，完成后建议代码后，建议、指导和所提供的信息真的很棒。” BP-5：“关于建议，这是我最喜欢，因为它确实帮助我更好地理解逻辑。” BP-6：“系统会告诉我们提交答案后我们错在哪里提交答案。”	BN-1：“系统会反复循环相同的题目次数太多。” BN-2：“我希望系统减少……的频率重复问题。”
C组 (ADVENTURE 3.1)	CP-1：“每完成练习后的反馈都非常有帮助。” CP-2：“我喜欢它能告诉我具体需要在哪里进行修改修正。” CP-3：“我觉得它看起来真的很好。它清楚地指出了程序认为有些地方应该修正或可能存在错误”.” CP-4：“总的来说，我喜欢有反馈可以帮助了解错误出在哪里is.” CP-5：“我喜欢它能告诉我哪些部分需要修改。”	CN-1：“这些问题往往重复得太频繁。” CN-2：“橙色按钮经常重复旧问题。”

注：使用缩写表示反馈类型和参与者组别。P = 积极反馈，N =消极反馈。A = A组学生，B =B组学生，C = C组学生。例如，AP表示A组学生给出的积极反馈，AN表示B组学生给出的消极反馈。

表4学习者对 ADVENTURE 的评论所反映的反馈类型。

反馈类型 反馈	示例 评论	推理
错误解释	BP-6、CP-2 至 CP-5	识别并解释学习者的编码错误。
代码建议	BP-3、BP-4、CP-5	提供建议或指导，以帮助实现正确逻辑或获得改进方案。
概念澄清	BP-5	支持概念理解，而非仅仅纠正错误。

注：此 示例评论摘自学习者的评论，见表 3。

差异体现在三项特征上：正确提交、错误提交，以及缺少必要逻辑的提交。A组的错误提交和缺少必要逻辑的提交显著多于B组和C组，而正确提交显著少于B组和C组。这些发现表明，使用生成式人工智能的参与者与单独使用自适应模式的学习者相比，使用这些模式的学习者编程效果更好。尽管B组和C组之间没有显著差异，但条形图显示，C组正确提交更多、缺乏逻辑的提交更少，跳过的问题也少于B组。这表明，与单独使用任一模式相比，生成式人工智能—自适应混合模式可能更有效地支持学习者提升编程技能。

关于学习者对生成式人工智能模型所生成反馈的看法，Yang 等人（2024年）提出了 CREF，该系统集成 GPT-4 来生成代码反馈，并证明在评估的 12 个大语言模型中，GPT-4 的正确率最高。本研究为考察学习者对反馈有用性的感知，向参与过两种生成式人工智能模式的学习者提出了以下问题：“您是否认同该反馈对您有帮助？”对学习者的回答进行分析，以回应第三个研究问题：“学习者如何看待生成式人工智能所提供反馈的有用性？”

图中结果图10 表明，B组和C组的大多数参与者都略微认同，生成式人工智能反馈有助于他们理解自己的代码提交。此外，表2中的结果表明，大多数参与者略微认同这两项系统功能——代码提交反馈和下一道练习推荐——易于使用且实用。然而，学习日志结果显示，生成式人工智能模式推荐的许多问题，参与者此前已经完成。同样，混合模式也经常推荐重复问题。学习者日志进一步显示，混合模式中的参与者更常选择自适应机制推荐的练习，而非生成式人工智能推荐的练习。这些发现表明，尽管参与者认为生成式人工智能生成的反馈有帮助，但他们往往更倾向于选择自适应机制推荐的练习。

6.2. 结论

总之，本研究结果表明，ADVENTURE 中的生成式人工智能—自适应混合模式有效支持了学习者练习编程并提升编程技能。该模式提供了灵活性，允许学习者选择由自适应方法或生成式人工智能模型推荐的下一项练习，同时提供人工智能生成的反馈，以帮助理解代码。此外，结果显示，混合模式的参与者完成正确提交的数量最多，且大多数人倾向于选择自适应方法推荐的练习。这些发现凸显了将自适应机制与生成式人工智能反馈相结合的价值；与单独采用任一模式相比，这种方式能更有效地支持编程教育。

6.3. 局限性

本研究有三项主要局限。第一项与生成式人工智能模型生成的练习推荐功能有关。尽管本研究使用的提示（如图A.1所示）要求 GPT-4 不要推荐先前对话或聊天历史中出现过的练习，但系统仍生成了一些重复内容，令学习者感到困惑。这一问题反映了大语言模型的一个已知局限：它们并不总能准确处理提示中的全部输入，尤其是在提示较长或包含复杂上下文信息时（Hosseini等人，2024）。尽管本研究并未旨在解决大语言模型的这一局限，为了减少学习者的困惑，参与者可以选择重复练习，或继续进行自适应机制生成的推荐。在本研究中，这种做法是生成式人工智能模式中的替代流程，也是自适应与生成式人工智能模式相结合时的主要流程。未来的工作将着重优化提示词结构，以减少大语言模型生成推荐时出现的此类不一致。第二项局限是实验组之间存在性别失衡，如第4.1节所述。由于开发该系统的目的是

支持初学者练习编程技能，我们招募了泰国一所大学信息技术专业的一、二年级本科生。然而，本研究的参与者人数有所减少：最初表示有意参加实验的143名学生中，只有123名（男性 = 107 和女性= 16）参与了实验，最终只有87名（男性 = 73 和女性 = 14）完成了实验和问卷。鉴于这一目标群体的特点，要在所有实验组中实现性别比例均衡颇具挑战。因此，本研究着重比较实验组，而非分析各组内部的性别差异。不过，我们计划在未来的工作中扩大参与者人数，以改善各组的性别均衡情况。第三项局限与问卷的翻译有关。本研究使用的问卷改编自（Wang 等人，2014、2020、2024年），原问卷以英语编写，后被翻译成泰语，以便在泰国一所大学对本科生开展实验。未来的研究需要审校泰语版本，确保译文条目的准确性和清晰度，并重新评估其信度和效度。

6.4. 未来工作

关于未来工作，为进一步提升 ADVENTURE 的有效性，我们根据本研究的发现提出了几项实施策略：(1) 扩展生成式人工智能生成的反馈，纳入对代码错误的自然语言解释，并提供将反馈翻译成其他语言的选项，以提升理解效果；(2) 开发以学习者为中心的分析仪表板，通过提供个人进度、优势和反复出现的错误的可视化洞察，促进自我调节学习。此外，我们计划招募来自多所大学的学习者，以扩大参与者群体并组建更加均衡的实验组。另外，我们还计划改进部分系统功能，例如避免推荐已完成的练习，并优化生成式人工智能模型所推荐练习的难度级别。此外，我们打算探索利用生成式人工智能进一步改善学习成果、支持技能发展的策略。

CRediT 作者贡献声明

Lalita Na Nongkhai： 撰写——审阅与编辑、撰写——初稿、可视化、验证、软件、方法学、调查、形式分析、数据整理、概念化。**Jingyun Wang：** 撰写——审阅与编辑、可视化、验证、监督、项目管理、方法学、形式分析、概念化。**Adam Wynn：** 撰写——审阅与编辑、验证、形式分析。**Takahiko Mendori：** 撰写——审阅与编辑、验证、监督、资源、项目管理、经费获取、概念化。

伦理审批与参与同意

本研究已获高知工科大学人体研究伦理委员会批准，并遵循相关指南和法规（方案编号：283；批准日期：2023年3月7日）。

材料与代码可用性

可通过以下网站访问 CONTINUOUS https://github.com/lalita-nk/CONTINUOUS.git。

伦理审批与参与同意

本研究已获高知工科大学人体研究伦理委员会批准，并遵循相关指南和法规（方案编号：283；批准日期：2023年3月7日）。所有参与者均已提供知情同意，并严格保护其隐私权。如需获取数据，可向通讯作者发送电子邮件提出申请。

经费

本研究由日本学术振兴会科学研究费助成事业资助，资助编号为 JP22K12318。

利益冲突声明

作者声明，他们不存在任何已知的竞争性财务利益或可能影响本文所报告工作的个人关系。

致谢

我们谨此感谢 ANGGUO ZHOU 先生在杜伦大学攻读硕士期间，为生成式人工智能集成功能提供支持。

附录

图 A.1 展示了用于 GPT-4 的提示，其中包含五个变量。这些变量包括 ：

(1) “question_content”：当前问题的内容。(2) “correct_code”：指导教师针对“question_content”提供的答案。(3) “submitted_code”：学习者提交的代码。(4) “chat_history”：GPT-4 为学习者生成的先前回复；(5) “context,“：GPT-4 为推荐的下一个问题提供的背景信息。

```
# main prompt
        combined_prompt = PromptTemplate(
            input_variables=["question_content", "correct_code", "submitted_code", "context", "chat_history"],
            template="""
            You are an instructor assisting a student in learning coding concepts.

            1. First, verify if the student's submitted code is correct based on the question_content and the correct_code provided:
                - question_content: {question_content}
                - Correct Code: {correct_code}
                - Submitted Code: {submitted_code}

            If the submitted code is incorrect:
                - Highlight the incorrect part of the code.
                - Show the correct code and explain the mistake. For example:

                Submitted Code:
                height = 7
                width = 5
                perimeter = 2 - (height + width) “The operation in the function is subtraction, not addition”
                print(perimeter)
                area = height * width
                print(area)

                the right part should be: perimeter = 2 * (height + width)

                Here is the correct code:
                height = 7
                width = 5
                perimeter = 2 * (height + width)
                print(perimeter)
                area = height * width

            2. Based on the previous conversation history:

                Chat History: {chat_history}

                explain why the student might have made this mistake and identify the knowledge points where the student may be struggling.

            3. Based on the student's performance, recommend a new exercise following the instructions below:

            - If the code is incorrect: Recommend exercises from the context provided below that address the specific mistake and expand the student's knowledge.
            - If the code is correct: Recommend more challenging exercises from the context provided below to further develop the student's understanding.

            Justify your recommendation by discussing the student's mastery of relevant knowledge points from previous conversations or Chat History.

            Warning:
            1.Do not recommend exercises that have appeared in previous conversations or Chat History.
            2.Do not include the 'code' content of the recommended exercise.

            Recommended Exercise:

            Question ID:
            Content:

            Recommended Reason:

            Note: The context provided below contains all exercises retrieved from the vector store to support recommendation.
            Context: {context}
            """
            )
```

图A.1. 与 GPT-4 配合使用的提示。

数据可用性

出于个人用户隐私考虑，本研究期间生成的原始学习者数据不公开。

参考文献

Azaiz, I., Kiesler, N., & Strickroth, S. (2024). 使用 GPT-4 为编程练习生成反馈。载于 *ACM* 计算机科学教育创新与技术会议论文集（*ITiCSE '24)*. https://doi.org/10.1145/3649217.3653594。意大利米兰。Chang, R.-C., & Zhang, J. (2024). CommunityKG-RAG：利用知识图谱中的社区结构，提升事实核查中的检索增强生成。*arXiv* 预印本 *arXiv:2408.08535*。 https://doi.org/10.48550/arXiv.2408.08535 Chase, H. (2022). LangChain：构建具有背景感知能力的推理应用。 https://github.com/langchain-ai/langchain。Chookaew, S., Panjaburee, P., Wanichsan, D., & Laosinchai, P. (2014). 一种个性化电子学习环境，旨在促进学生’ 对计算机基础知识的概念学习

编程。*Procedia - Social and Behavioral Sciences, 116*，815–819。https://doi.org/10.1016/j.sbspro.2014.01.303 Deeva, G., Bogdanova, D., Serral, E., Snoeck, M., & De Weerdt, J. (2021). 面向学习者的自动化反馈系统综述：分类框架、挑战与机遇。*Computers & Education, 162*，文章 104094。https://doi.org/10.1016/j.compedu.2020.104094 Derus, S. M., & Ali, A. Z. M. (2012). 编程学习中的困难：学生的观点。第一届教育当前问题国际会议（*ICCIE 2012*）论文集。https://doi.org/10.13140/2.1.1055.7441。印度尼西亚日惹。Djambong, T., & Freiman, V. (2016). 学生通过可视化编程或实体编程环境培养的 计算思维技能的基于任务的评估。第十三届国际信息社会发展协会数字时代认知与探索性学习国际会议（*CELDA*）论文集。德国曼海姆 https://eric.ed.gov/? id=ED571389。Dong, C., Yuan, Y., Chen, K., Cheng, S., & Wen, C. (2023). 如何利用知识图谱增强检索增强生成（KG-RAG）为任意课程构建自适应人工智能导师。*arXiv* 预印本 *arXiv:2311.17696*。https://doi.org/10.48550/ arXiv.2311.17696

Fabic, G. V. F., Mitrovic, A., & Neshatian, K. (2018). 移动 Python 辅导系统中的自适应问题选择。第 26 届用户建模、自适应与个性化会议（UMAP '18）附属出版物，269–274。https://doi.org/10.1145/ 3213586.3225235。ACM。Frankford, E., Antensteiner, T., Vierhauser, M., Sauerwein, C., Wallner, V., Groher, I., Pl¨osch, R., & Breu, R. (2025). 自动化编程评测系统中的反馈类型综述。ACM《计算机教育汇刊》。在线优先出版。https://doi.org/10.1145/3773911 Gabbay, H., & Cohen, A. (2024). 在编程大规模开放在线课程中结合大语言模型生成的反馈与基于测试的反馈。载于第十一届 ACM 大规模在线学习会议论文集（L@S '24）（第 177–187 页）。ACM。https://doi.org/10.1145/3657604.3662040。Gan, W., Qi, Z., Wu, J., & Lin, J. C.-W. (2023). 教育中的大语言模型：愿景与机遇。2023 IEEE 大数据国际会议（BigData），4776–4785。发表于 2023 IEEE 大数据国际会议（BigData）。https://doi.org/10.1109/bigdata59044.2023.10386291。意大利索伦托。Garcia, M. B. (2025). 使用 ChatGPT 教授和学习计算机编程：生成式人工智能技术兴起背景下的文献快速综述。《教育与信息技术》。在线优先出版。https://doi.org/10.1007/s10639-025-13452-5 Gavrilovi´c, N., Arsi´c, A., Domazet, D., & Mishra, A. (2018). 用于实现自适应学习过程并提升学习者 Java 编程语言技能的算法。《工程教育中的计算机应用》，26(5)，1362–1382。https://doi.org/ 10.1002/cae.22043 George, S. D., & Dewan, P. (2024). NotebookGPT——在编程练习中促进并监控学生明确提出的轻量级 GPT 求助请求。载于第 29 届智能用户界面国际会议（IUI '24 companion）配套论文集（第 62–65 页）。ACM。https://doi.org/10.1145/3640544.3645234。Hare, R., & Tang, Y. (2024). 面向个性化学生支持的本体驱动强化学习。2024 IEEE 系统、人类与控制论国际会议（SMC），2555–2560。发表于 2024 IEEE 系统、人类与控制论国际会议（SMC），马来西亚古晋。https://doi.org/10.1109/ smc54092.2024.10832036 Hosseini, P., Castro, I., Ghinassi, I., & Purver, M. (2024). 大语言模型一个令人费解的失效问题的高效解决方案：长上下文窗口并不意味着大语言模型能够完美分析长序列。https://arxiv.org/abs/2408.01866。Kazemitabaar, M., Ye, R., Wang, X., Henley, A. Z., Denny, P., Craig, M., & Grossman, T. (2024). CodeAid：评估一种基于大语言模型的编程助手在课堂中的应用，该助手兼顾学生与教育者的需求。载于 2024 年计算机系统中的人为因素CHI会议论文集。计算机协会。https://doi.org/10.1145/3613904.3642773。文章 650，共 20 页）。Kosar, T., Ostoji´c, D., Liu, Y. D., & Mernik, M. (2024). ChatGPT 时代的计算机科学教育：编程初学者课程实验的经验。《数学》，12(5)，629。https://doi.org/10.3390/math12050629 Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Kuttler, H., Lewis, M., Yih, W., Rocktaschel, T., Riedel, S., & Kiela, D. (2020). 面向知识密集型自然语言处理任务的检索增强生成。载于 H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin（编），《神经信息处理系统进展》，33，第 9459–9474 页）。Li, Q., Fu, L., Zhang, W., Chen, X., Yu, J., Xia, W., Zhang, W., Tang, R., & Yu, Y. (2024a). 面向教育的大语言模型适配：基础能力、潜力与挑战。https://doi.org/10.48550/ARXIV.2401.08664 Li, Y., Yang, R., Zhang, X., Shi, P., Yang, D., & Huang, X. (2024b). ProgMate：基于大语言模型的智能编程助手。载于 2024 IEEE 教育前沿会议（FIE）论文集。IEEE。https://doi.org/10.1109/FIE61694.2024.10893312。Majumdar, R., Prasad, P., & Sane, A. (2024). 利用 Chat-GPT 挖掘编程问题求解中的认知行动。载于 B. Paaßen, & C. D. Epp（编），第 17 届教育数据挖掘国际会议（EDM 2024）论文集（第 628–633 页）。国际教育数据挖掘学会。https://doi.org/10.5281/ zenodo.12729902。

Na Nongkhai, L., Wang, J., & Mendori, T. (2022). 从计算思维教育的视角构建多种编程语言本体。载于《数字时代认知与探索性学习国际会议第19届会议论文集》（CELDA 2022）（第66–72页）。https://doi.org/10.33965/celda2022_ 202207l009 Na Nongkhai, L., Wang, J., & Mendori, T. (2023). 基于多种编程语言本体的自适应学习支持系统。载于 J. L. Shih（编），《第31届计算机教育国际会议论文集》（第141–143页）。https://doi.org/10.58459/icce.2023.958。亚太计算机教育学会。Na Nongkhai, L., Wang, J., & Mendori, T. (2025). 基于多种编程语言本体的自适应学习支持系统的开发与评估。《教育科学》，15(6)，724。https://doi.org/10.3390/ educsci15060724 OpenAI. (2023). GPT-4技术报告。https://doi.org/10.48550/arXiv.2303.08774。Pel´anek, R. (2016). Elo 评级系统在自适应教育系统中的应用。《Computers & Education》，98，169–179。https://doi.org/10.1016/j. compedu.2016.03.017 Shute, V. J. (2008). 聚焦形成性反馈。《教育研究评论》，78(1)，153–189。https://doi.org/10.3102/0034654307313795 Tang, X., Wong, S., Huynh, M., He, Z., Yang, Y., & Chen, Y. (2024). SPHERE：通过对大语言模型输出进行结构化审查，在编程课堂中扩展个性化反馈。载于《2024年计算机系统中的人为因素CHI会议论文集》（CHI '24）。计算机协会。https://doi.org/10.48550/ arXiv.2410.16513。Wang, J., Mendori, T., & Xiong, J. (2014). 使用以课程为中心的本体的语言学习支持系统及其评估。《Computers & Education》，78，278–293。https://doi.org/10.1016/j.compedu.2014.06.009 Wang, J., Shimada, A., Oi, M., Ogata, H., & Tabata, Y. (2020). 用于支持有意义电子书学习的可视化系统的开发与评估。《交互式学习环境》，31(2)，836–853。https://doi.org/10.1080/ 10494820.2020.1813178 Wang, J., Wynn, A., Mendori, T., & others. (2024). 促进有意义语法学习的主题图式学习管理系统：以日语语法学习为例。《智能学习环境》，11，53。https://doi.org/10.1186/ s40561-024-00338-1 Xu, H., Gan, W., Qi, Z., Wu, J., & Yu, P. S. (2024a). 教育领域的大语言模型：综述。arXiv预印本 arXiv:2405.13001。https://doi.org/10.48550/ arXiv:2405.13001 Xu, Z., Cruz, M. J., Guevara, M., Wang, T., Deshpande, M., Wang, X., & Li, Z. (2024b). 利用知识图谱进行检索增强生成，以实现客户服务问答。《第47届ACM信息检索研究与发展国际会议论文集》，33，2905–2909。发表于 SIGIR 2024：第47届ACM信息检索研究与发展国际会议。https://doi.org/10.1145/3626772.3661370。美国华盛顿特区。Yang, B., Tian, H., Pian, W., Yu, H., Wang, H., Klein, J., Bissyand´e, T. F., & Jin, S. (2024). CREF：面向编程辅导教师的基于大语言模型的对话式软件修复框架。载于《第33届ACM SIGSOFT国际软件测试与分析研讨会论文集》（ISSTA '24），2024年9月16–20日。奥地利维也纳：ACM。https://doi.org/10.1145/3650212.3680328。Zhang, J., & Liu, K. (2024). JavaLLM：用于Java编程教育的微调大语言模型。载于《第8届国际计算机科学与智能控制研讨会论文集》（ISCSIC）（第276–280页）。IEEE。https://doi.org/10.1109/ISCSIC64297.2024.00064。Zheng, W., Du, Q., Fan, Y., Tan, L., Xia, C., & Yang, F. (2022). 基于知识结构树的个性化编程练习推荐算法。《智能与模糊系统杂志》，42(3)，2169–2180。https://doi.org/ 10.3233/JIFS-211499