AISLI

生成式人工智能辅助编程与儿童青少年计算思维发展研究综述

需求表达、逻辑组织、人机协作与成果验证

GENERATIVE AI-ASSISTED PROGRAMMING AND
COMPUTATIONAL THINKING DEVELOPMENT
IN CHILDREN AND ADOLESCENTS

AISLI 研究综述 / AI 生成扩充工作版
2026 年 9 月 11 日

编制与 AI 生成说明

本报告由 AISLI 使用人工智能（AI）生成，依据公开学术文献与相关原始资料编制。AI 参与了资料检索辅助、文献归纳、框架构建、正文撰写、分析图表制作和文档排版。本版本为扩充工作版，供研究讨论与后续审阅使用。

报告采用批判性叙述综述方法，结合 Sider Scholar 学术检索、出版机构原始页面、作者公开稿及公开研究资料进行定向核验。资料核验截止 2026 年 9 月 11 日。全文、摘要和题录层面的可得性限制在相关位置说明；未经核对的样本与效果不补写为事实。本文未开展穷尽性系统筛选、预注册、独立双人编码或元分析。

正文区分原研究发现、综合分析与本文提出的教学或研究设计。统计数据和具体研究使用脚注与参考文献标示来源；新绘分析图与表格属于 AISLI 综合提出的框架，不能作为实测结果。真实照片保留图源与版权说明，AI 概念插画明确标识，不对应真实研究现场。

AI 生成与自动化校核不能替代领域专家的学术判断。本次修订完成了资料对照、文字与结构检查及导出版面核查，但不据此声称已经通过独立同行评审或正式发布审批。涉及具体课程、研究与机构决策时，应结合原始文献及相应情境继续审阅。

编制主体：AISLI
版本性质：AI 生成扩充工作版
语言：中文；英文与西文数字采用 Times New Roman
版本日期：2026 年 9 月 11 日

摘要

生成式人工智能辅助编程涵盖借助生成模型解释代码、提供反馈、生成程序片段、辅助调试和开展项目创作等活动。vibe coding 是其中需要专门辨析的一类实践，不等同于全部 AI 辅助编程。对儿童青少年而言，生成工具可能降低语法与环境配置门槛，使更多学习者进入设计和创作；也可能使问题分解、算法理解、调试与验证被外包。本综述区分“作品完成”“有工具问题求解”和“计算思维发展”，分析需求表达、逻辑组织、人机协作与成果验证如何成为有实质学习价值的活动。

目前已有儿童青少年使用 AI 代码生成及 LLM 协作编程的直接研究，也出现了明确以儿童 vibe coding 为对象的研究。然而，样本、教学安排、工具形态和测量方式差异很大，长期保持与跨情境迁移的证据仍有限。大学生研究揭示的写作、计算机知识、元认知与人机互动关系，为设计提供邻近证据，不能直接证明同样的训练对儿童有效。

本文提出“表述—分解—协作—验证—解释与迁移”的学习循环，并给出任务设计、渐退支持、过程证据与评价量规建议。核心判断是：自然语言可以成为计算表达的新入口，但计算思维的发展依赖学习者是否持续承担建模、判断和验证责任。课程应同时评价作品、过程与撤除 AI 后的能力，避免以生成速度、提示次数或作品复杂度替代学习证据。

关键词：生成式人工智能辅助编程；自然语言驱动的编程学习；计算思维；儿童青少年；生成式人工智能；需求表达；问题分解；人机协作；程序验证；学习迁移

目录

按研究问题组织正文，附方法说明与参考文献。

图表目录

图题置于图下，表题置于表上。概念插画不计入图号。

图表目录（续）

1 问题界定与证据范围

教育技术研究应先界定学习者在做什么，再讨论工具名称。本文以生成式人工智能辅助编程作为研究对象，并将自然语言驱动的生成与迭代实践置于这一范围中讨论。并非所有 AI 辅助编程都属于同一种教学方法：解释一个错误、补全一行代码、生成整个应用，以及由智能体自行实现项目，涉及不同程度的认知外包和控制权分配。

本文关注儿童青少年计算思维发展，核心结果包括抽象与建模、问题分解、算法表达、调试与验证、迁移以及对计算系统的批判性理解。需求表达与协作被视为可能促进这些结果的学习过程，不自动等同于计算思维本身。成年开发者的生产效率、大学生的任务成绩和儿童的能力发展分别讨论。

采用定向检索与关键文献追溯的批判性叙述综述，资料截至 2026 年 9 月 11 日。通过 Sider Scholar 的 OpenAlex 检索、ACM 与期刊原始页面、作者公开稿和 arXiv 定位资料，检索词包括 vibe coding children、AI code generation novice programming、computational thinking generative AI、prompting children、human AI collaboration 和 code verification learning。经典计算思维框架用于概念界定，近年实证用于分析新的工具情境。

纳入资料分为儿童青少年直接证据、大学或成人邻近证据、理论与设计框架三类；预印本与仅能核对题录或摘要的资料另行标示。未进行穷尽性数据库筛选、双人独立编码、预注册和元分析，因此不声称已经证明总体效应。证据比较保留年龄、教学活动、工具控制、研究设计和测量时点，避免把不同研究中的百分比直接合并。

表 01 证据范围与儿童发展结论的关系

证据层次	主要用途	解释边界
儿童青少年直接研究	观察特定教学与学习结果	不能自动跨年龄课程外推
大学或成人邻近研究	提出能力与交互机制线索	不能当作儿童因果证据
概念与设计框架	组织课程与研究假设	效果仍待检验
题录或摘要级资料	识别研究主题与待补资料	不使用未核验全文细节

注：本文证据分类。

1.1 从流行实践名称转向可研究的学术对象

本报告采用“生成式人工智能辅助编程”作为上位概念，英文为 Generative AI-assisted Programming。这一名称能够覆盖解释代码、提供反馈、生成片段、辅助调试以及自然语言驱动的项目开发，同时容纳不同的人机责任分配。vibe coding 保留为其中一种值得专门考察的实践形态，尤其涉及凭借运行效果和自然语言反馈持续迭代、对实现细节审查程度不一的开发方式。两者并非可以在所有语境中互换的同义词。

研究对象的边界会改变结论。学生先画算法、再请求语法提示，与学生提交一句愿望后接受完整应用，虽然都使用生成模型，却经历了不同的学习活动。如果将它们统一记为“使用 AI”，研究就难以解释效果为何不同。本报告建议同时记录任务控制、代码生成范围、解释责任和验证方式；工具品牌只是环境信息，不能替代教学活动的描述。

“自然语言驱动的编程学习”适合描述具体交互入口，“人机协同程序设计”强调共同完成任务的组织形式，“生成式人工智能辅助编程”则适合作为综述题名中的研究领域名称。选择后者并不预先认定学习者已经实现了高质量协作。协作是否成立，仍需要从学生的目标设定、判断、修改与验证行为中寻找证据。

表 02 三个相关术语的研究范围

术语	适用范围	使用时的限定
生成式 AI 辅助编程	解释、反馈、生成、调试等活动	作为本综述的上位概念
自然语言驱动的编程学习	以自然语言表达需求与迭代	描述交互方式，不预设学习效果
vibe coding	依赖生成与运行反馈的开发实践	需报告审查程度与责任分配

注：AISLI 的概念界定；不作为领域统一术语标准。

1.2 将研究问题限定在可观察的变化上

本文把问题进一步分解为：哪些计算活动被保留、被增强或被外包；哪些学生在什么支持下受益；所得变化能否在撤除 AI 后保持；评价是否受到语言表达、既有编程经验和工具熟悉度影响。这些问题比直接询问“新方式好不好”更有解释力，也允许研究发现积极效果与限制同时存在。

在证据提取时，应将作者报告的结果、本文对研究的解释以及本文提出的设计建议分别呈现。小学研究不能代表所有未成年人；大学课程经验可以帮助发现机制，却需要适龄研究验证。本文采用这种分层处理方式，因此保留有关 vibe coding 的原始论文名称与检索词，而不通过更名抹去其研究情境。

2 计算思维在自然语言编程时代的连续性与变化

Wing 将计算思维作为运用计算机科学基本概念解决问题、设计系统和理解行为的一种能力主张。这一视角并未把它限定为手工输入语法。AI 降低代码书写门槛后，教育仍需要回答学习者是否能把模糊问题转化为可执行、可检查的表示，以及能否理解表示的限制。[1]

图 01 完成计算机任务的学习者（2010 年）

注：历史小学教学情境；不对应 vibe coding 研究或 AI 代码生成任务。 摄影或版权：woodleywonderworks；CC BY 2.0[2]；原始图片[3]。按原比例排版。

1 Wing (2006)。文献[1]。原始资料

2 CC BY 2.0。原始资料

3 原始图片。原始资料

概念插画 | AI 生成；不对应真实学习者、学校或研究现场。

Brennan 与 Resnick 提出的概念、实践和视角框架，使评价超出最终代码：循环、事件和条件等概念，与迭代、调试、复用和抽象等实践，以及表达、连接和质疑等视角相互关联。生成式 AI 辅助编程可能让学习者更快创作，却不能以创作热情替代对条件和状态的理解。过程访谈、项目演示和修改任务仍然有价值。[4]

Grover 与 Pea 对 K—12 计算思维研究的讨论提醒教育者，概念定义、教学设计和能力测量需要同时推进；Weintrop 等将计算思维扩展到数据、建模与仿真、计算问题求解和系统思维。AI 时代的变化并非删除这些能力，而是把更多学习机会放在任务形式化、证据检查与系统边界判断上。[5, 6]

4 Brennan & Resnick (2012)。文献[2]。原始资料

5 Grover, Shuchi; Pea, Roy (2013)。文献[3]。原始资料

6 Weintrop et al. (2016)。文献[4]。原始资料

图 02 自然语言入口与计算思维的连接

注：本文综合计算思维框架；表达方式不等于能力本身。[7, 8, 9]

必须避免两种概念膨胀。第一，把任何清楚表达都称为计算思维。优秀叙述有助于沟通，但只有涉及明确对象、关系、约束、过程或可执行规则时，才与计算建模产生直接联系。第二，把任何与 AI 对话都称为人机协作。协作应包含共同任务、责任分配、反馈和结果核验，而不仅是反复请求“再改好一点”。

自然语言是一种强大但含混的媒介。“做一个公平的抽签工具”没有说明谁有资格、是否允许重复、随机结果如何保存、怎样核查偏差。把这些问题逐项说明，才把生活意图推进为可计算规则。课程的重点可以从“会不会写某个语法”扩展为“能否明确表示问题”，但仍需适当的代码、流程图和数据表示帮助学生理解执行机制。

2.1 计算思维的表达形式变化与概念连续性

代码是一种表示，流程图、状态表、图块和自然语言规则也可以承载计算关系。判断学习者是否形成理解，不能只看他采用哪一种媒介，而应看不同表示之间能否相互解释。例如，学生说“如果所有人都借完了，就停止提醒”，还需要说明“所有人”和“借完”在系统中如何表示，何时检查条件，以及发生新增记录时规则是否仍然成立。

表示之间的转换提供了重要的教学机会。教师可以给出一段自然语言需求，让学生画出流程；也可以展示流程图，请学生找出生成代码中相应的条件。转换发生困难时，

7 Wing (2006)。文献[1]。原始资料

8 Brennan & Resnick (2012)。文献[2]。原始资料

9 Weintrop et al. (2016)。文献[4]。原始资料

问题可能源于概念尚未理解，也可能源于表达工具不熟悉，不能只根据单次书面解释作结论。允许口述、操作演示与图示相互补充，有助于提高评价的有效性。

2.2 认知外包与学习责任的重新分配

Risko 与 Gilbert 关于认知外包的分析，为理解外部工具如何改变任务负担提供了理论起点。把信息保存到外部载体或者借助工具完成部分加工，可能释放注意资源；但释放的资源是否用于理解，取决于任务安排和学习者的监控。该理论不是儿童使用生成式 AI 的效果证明，而是提出可检验机制的依据。[10]

在辅助编程中，减少记忆语法的负担可能让学生更多比较算法，也可能使其跳过算法讨论。两种路径都合理，不能仅凭“认知负荷降低”判断学习必然改善。需要进一步观察学生把时间用于什么活动，能否发现错误，以及支持减少后是否仍能完成相关推理。负荷问卷、过程记录和独立任务应结合解释，而不能互相替代。

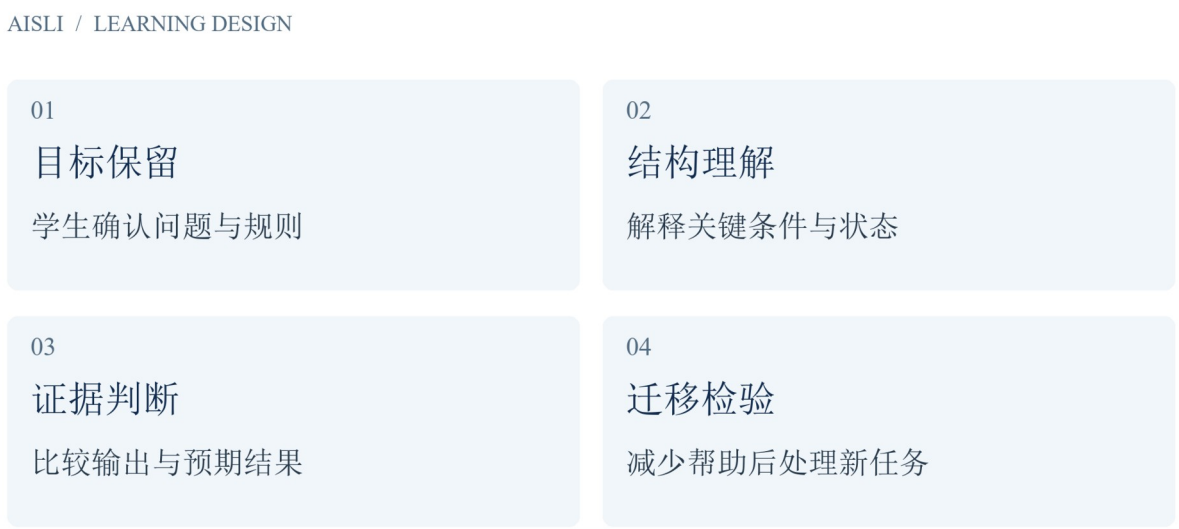

图 03 认知外包之后：学习责任的四个检查点

注：AISLI 综合提出的分析框架；不是认知负荷或学习效果的实测结果。参考认知外包与计算思维研究。[11, 12]

因此，课程目标可以从“每一行都由学生输入”转向“关键计算关系由学生理解并承担判断责任”。但这不意味着取消语法学习。适量的手工实现和代码阅读，仍可帮助

10 Risko, Evan F.; Gilbert, Sam J. (2016)。文献[5]。原始资料

11 Risko, Evan F.; Gilbert, Sam J. (2016)。文献[5]。原始资料

12 Wing (2006)。文献[1]。原始资料

学生建立对变量、条件、循环和数据变化的认识。何时借助生成、何时要求自己完成，应由当前学习目标与已有能力决定。

3 直接研究能够支持哪些判断

Kazemitabaar 等在 2023 年研究 69 名 10—17 岁编程初学者使用 AI 代码生成器，设置代码编写与人工修改任务。使用生成器提高了部分任务的完成与得分，研究没有发现所测人工修改能力下降；一周后的学习测验差异并未达到统计显著。它支持在具体设计下研究 AI 辅助入门，不支持“所有自然语言生成方式都提高长期计算思维”的结论。[13]

Yan 等 2025 年研究 82 名六至七年级学生的 LLM 支持协作编程，使用来自 Bebras 题目的计算思维前后测，报告实验组计算思维改善和较低认知负荷，自我效能组间差异不显著。该课程包含教师教学、同伴协作和结构化支持，不能将结果归于模型本身。论文对研究设计及四周、五周的时间描述存在不一致，本文据此保留方法报告的不确定性，不进一步计算统一效应量。[14]

2026 年 CHI 收录了 Si 等明确讨论儿童 vibe coding 学习与实践的研究，说明这一主题已经进入直接研究范围。本文核对了其正式题录与出版信息，但未取得可逐页核查的原始全文，因此不使用其详细样本、年龄、效果大小或具体观察作为已核验事实。后续更新应优先获得原文，完整提取教学任务、工具形态和研究限制。[15]

这组资料形成的判断是有条件的：当 AI 嵌入教师组织、同伴互动、解释和验证活动时，可能改善某些编程学习结果；当研究对象和评价不同，效果也可能不同。现阶段不能把“有一项积极研究”推论为普遍课程建议，也不能因为担忧依赖而否认所有受控使用的学习价值。

13 Kazemitabaar, Majeed; Chow, Justin; Ma, Carl Ka To; et al. (2023)。文献[6]。原始资料

14 Yan, Yi-Miao; Chen, Chuang-Qi; Hu, Yang-Bang; et al. (2025)。文献[7]。原始资料

15 Si, Janice Jianing; Li, Donglin; Wang, Qiuning; et al. (2026)。文献[8]。原始资料

表 03 儿童青少年直接研究的关键比较

研究	对象与安排	结果及限制
Kazemitabaar 等 2023	69 名 10—17 岁初学者；AI 代码生成	部分任务改善；所测修改能力未下降；一周测验差异不显著
Yan 等 2025	82 名六至七年级；教师与同伴协作	计算思维改善；自我效能差异不显著；设计与时长报告有不一致
Si 等 CHI 2026	儿童 vibe coding 主题研究	正式题录已核对；原始全文未获得，不使用样本或效果细节

注：原始资料与可得性边界。[16] [17] [18]

尤其需要识别学习测量的层次。编程任务完成率衡量当时解决任务的能力；没有 AI 时的修改任务更接近独立理解；延迟测验涉及保持；陌生问题中的规则迁移则涉及更广泛发展。每个层次都重要，但必须说明研究实际测量了哪一种，而不能统一写成“提升计算思维”。

3.1 新增小学与高中研究的证据位置

Xu 等 2026 年发表的小学编程研究涉及 71 名六年级学生，比较两个既有班级中的自我调节编程与 ChatGPT 支持条件，持续十周。研究结合学习结果、交互记录和访谈，报告了编程知识等方面的积极发现。这里的 AI 通过面向任务的界面参与解释与反馈，不能等同于不受约束的完整项目生成；两个班级的比较也不能当作 71 名学生分别随机分组。[19]

Guo 等 2026 年的高中研究涉及 83 名十年级学生，采用两个既有班级、八周教学和 Bebras 题目测量，报告 AI 支持组的计算思维总分变化更大。研究安排了流程图、代码理解、修改与调试，并限制直接索取整项作业的答案。因此，它更接近结构化的生成式 AI 辅助编程证据，而不是低审查程度的 vibe coding 证据。单校、班级层面的分组仍限制了因果解释和外推。[20]

16 Kazemitabaar, Majeed; Chow, Justin; Ma, Carl Ka To; et al. (2023)。文献[6]。原始资料

17 Yan, Yi-Miao; Chen, Chuang-Qi; Hu, Yang-Bang; et al. (2025)。文献[7]。原始资料

18 Si, Janice Jianing; Li, Donglin; Wang, Qiuning; et al. (2026)。文献[8]。原始资料

19 Xu, Jie; Sun, Dan; Feng, Yue; Shadiev, Rustam; Li, Yan (2026)。文献[9]。原始资料

20 Guo, R.; Li, G.; Miao, H.; Pi, Z.; Xie, L. (2026)。文献[10]。原始资料

Liu 等关于小学编程元认知支架的研究，比较自适应支架、预先安排的支架和控制条件。本文目前依据可核查的正式摘要使用这一资料，将其作为支架类型值得比较的研究线索；不据此补写未核对的年龄、效应量或课堂操作细节。摘要报告的积极结果不能代替全文方法审查。[21]

表 04 新增基础教育研究的证据位置

研究	研究对象与条件	可支持的判断与限制
Xu 等，2026	71 名六年级学生；两个班级；十周	结构化编程支持；不能等同于自由生成
Guo 等，2026	83 名十年级学生；两个班级；八周	报告计算思维测验改善；外推受限
Liu 等，2026	小学编程中的不同元认知支架	本文按摘要级证据使用，不补写细节

注：根据原始资料概括；不同研究的条件和结果不能直接合并。[22] [23] [24]

3.2 阅读积极结果时需要保留的条件

这几项研究共同提示，直接证据的对象正在从大学初学者扩展到基础教育，但“基础教育已有研究”与“普遍有效”之间仍有很长的论证距离。比较研究时至少要区分教师引导、对话限制、同伴安排、任务类型和测量工具。研究中的改善可能来自这些因素共同形成的学习环境，不宜将全部效果归给模型参数或自然语言交互本身。

还应检查前测的解释。组间差异未达到显著水平，并不证明两组完全等同；班级文化、教师期待、课外经验和设备使用机会仍可能不同。延迟测验、跨任务评价与多校重复研究，可以逐步补充这些局限。本文据此将现有结果表述为“在特定课程条件下具有积极证据”，而不把它提升为所有年龄、所有工具和所有教学方式的通用结论。

21 Liu, Jinfang; Zhang, Yi; Li, Wei; Wang, Qiyun; Niu, Pingxiu; Zhang, Xue (2026)。文献[11]。原始资料

22 Xu, Jie; Sun, Dan; Feng, Yue; Shadiev, Rustam; Li, Yan (2026)。文献[9]。原始资料

23 Guo, R.; Li, G.; Miao, H.; Pi, Z.; Xie, L. (2026)。文献[10]。原始资料

24 Liu, Jinfang; Zhang, Yi; Li, Wei; Wang, Qiyun; Niu, Pingxiu; Zhang, Xue (2026)。文献[11]。原始资料

4 邻近研究带来的机制线索

2026 年一项关于 vibe coding 技能的预注册研究考察 100 名高等教育阶段参与者，发现计算机科学成绩和写作能力与任务表现存在相关。控制一般认知测验后，计算机科学成绩的偏相关约为 0.281，写作能力约为 0.186，后者未达到常用显著性阈值。这说明表达与学科知识值得同时研究，但不能证明训练写作会导致 vibe coding 或儿童计算思维提升。[25]

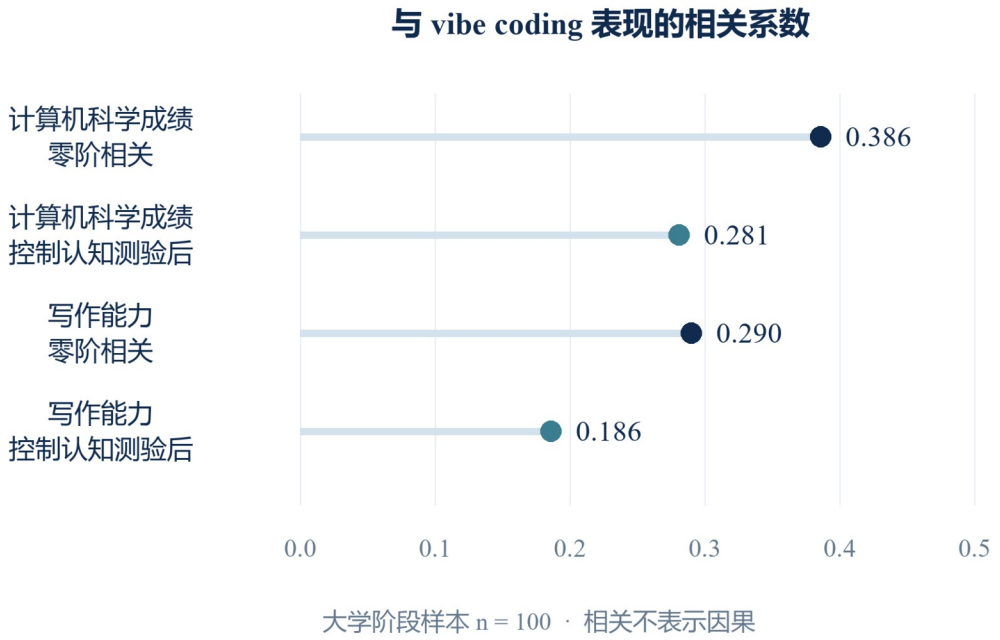

图 04 大学阶段研究中的能力相关及控制后变化

注：Thorgeirsson 等，2026，表 3；n=100。数值取三位小数；未绘置信区间，控制后写作相关 p=0.066。不可外推为儿童训练的因果效果。[26]

这一区分直接影响教育建议。课程可以有理论依据地设计更明确的需求表达活动，却应把“该活动是否改善计算思维”作为待检验问题。不能将成人横断面相关倒置为儿童发展因果，也不能把控制变量后的不显著解读为写作完全没有作用。研究样本、任务及统计不确定性共同限制解释。

Prather 等关于初学者使用生成式 AI 的质性研究指出，原有能力和元认知会影响学习者如何利用工具：一些学生用 AI 支持理解，另一些可能形成能力错觉。该研究有助

25 Thorgeirsson, Sverrir; Weidmann, Theo B.; Su, Zhendong (2026)。文献[12]。原始资料

26 Thorgeirsson, Sverrir; Weidmann, Theo B.; Su, Zhendong (2026)。文献[12]。原始资料

于提出分层支持和过程观察的设计问题，但质性样本不用于估计儿童中某种行为的普遍发生率。[27]

CodeAid 在大学课程中探索以解释、伪代码和有限支持代替直接给出完整解答，说明工具可以通过交互约束塑造学习活动。这类部署研究提供可实现的设计参考，尚不足以证明相同限制对所有年龄和先备水平最优。教师需要根据任务选择何时解释、何时提示、何时要求学生先尝试。[28]

初学者与代码模型相互误读的研究进一步显示，即使学习者知道自己想达到什么结果，也可能难以用模型能够可靠执行的形式表达和修订要求。错误既可能来自学生对计算概念理解不足，也可能来自模型误解和交互反馈不清楚。课程不应把一切失败归为学生“不会提问”。[29]

关于 vibe coding 与软件工程的早期预印本观察到快速原型与工程质量之间的张力。这些成人或大学生情境提醒课程加入测试、可维护性和版本管理，但尚不能推算儿童使用后的发展轨迹。把它们用于提出机制假设，比将其写成已确证的儿童教育结论更合适。[30，31]

4.1 从成人经验提出机制，而非直接移植结论

大学生或成年初学者通常具有更成熟的阅读、抽象表达与自我管理能力，任务目标也可能更接近完成课程作业或提高开发效率。儿童则可能仍在学习如何描述因果、保持规则一致和判断信息可靠性。同样一句“请解释错误”，对不同学习者提出的要求并不相同。邻近研究的价值在于发现可能的困难与支持方式，不能用年龄标签简单换算效果。

一个值得检验的机制是先备知识与反馈可理解性之间的关系。已有概念基础的学生可能从模型的简短提示中定位问题；缺少基础的学生即使获得较长解释，也可能不知道哪一句与自己的错误有关。因此，所谓个性化不能只根据学生是否继续提问来判断，还应结合其预测、解释与修改行为，确认提示是否进入了理解过程。

另一个机制是流畅表达带来的可信错觉。生成代码具有完整缩进和熟悉命名，解释使用专业术语，都会使输出显得可靠。但呈现流畅与逻辑正确是不同属性。课堂可通过

27 Prather, James; Reeves, Brent; Leinonen, Juho; et al. (2024)。文献[13]。原始资料

28 Kazemitabaar, Majeed; Ye, Runlong; Wang, Xiaoning; et al. (2024)。文献[14]。原始资料

29 Nguyen, Sydney; Babe, Hannah McLean; Zi, Yangtian; et al. (2024)。文献[15]。原始资料

30 Gama, Kiev; Calegario, Filipe; Jackson, Victoria; et al. (2025)。文献[16]。原始资料

31 Pimenova, Veronica; Fakhoury, Sarah; Bird, Christian; et al. (2025)。文献[17]。原始资料

比较两个看似合理、实际行为不同的短程序，引导学生寻找区分它们的输入。这里的目标是训练证据判断，不能把“怀疑所有输出”本身当作能力。

4.2 机制分析需要明确替代解释

如果使用 AI 的学生更愿意反复修改，可能因为即时反馈降低了挫败，也可能只是生成成本低而不断尝试。前一种解释强调持续投入，后一种可能缺少明确假设。仅统计修改次数无法区分两者；需要查看每次修改之前是否有预期、修改之后是否有核查，以及学生能否解释保留某个版本的理由。

同样，学生对工具满意，不必然说明知识掌握更好；表现暂时下降，也不必然说明支架无效，可能反映任务变难或学生开始承担更多判断。研究报告应把使用体验、即时表现、学习保持与迁移分别列出，并允许结果方向不同。这种处理能够避免把技术研究写成产品评价，也能给课程改进留下更具体的依据。

5 清晰表达需求怎样进入计算建模

需求表达的教育价值，在于让学生把尚未成形的想法变成别人能够检查的规则。可以从使用者、目标、输入、输出、约束和验收条件六个方面展开。“做一个有趣的小游戏”是创作愿望；“角色碰到障碍后回到起点，保留已经获得的分数，每轮最长一分钟”则包含状态变化、条件和时间约束。

需求并非越长越好。冗长叙述可能夹杂互相冲突的条件，儿童也可能复制成人化的提示模板而不理解。好的课堂活动应使每一句需求都有用途，能够举例说明，并允许发现问题后修订。学生可以用图、口述、卡片和演示表达，不应以写作长度将语言能力差异误当作计算思维差异。

图 05 从创作愿望到可以检查的需求

注：本文教学设计建议；可通过口述、图示或文字表达。

一个可实施的例子是设计图书借阅提醒。学生先解释谁借书、何时归还、怎样记录；随后讨论同一天借多本书、记录为空、归还日期修改和重复提醒等情况。这些边界讨论引出数据结构、条件判断和状态管理。AI 可以提出遗漏的情况，但教师应要求学生判断哪些与任务有关，并说明自己的规则。

需求还需要可反驳性。“界面要好看”“运行要正确”不能直接验收；“输入空值时显示说明，不创建借阅记录”可以通过测试检查。将质量愿望转换为可观察条件，是逻辑组织与成果验证之间的桥梁。这个转换过程应保留学生自己的草稿，使教师能够看到要求怎样变得清楚。

本文建议采用先表示、后生成、再解释的顺序。学生先画出对象与关系或写出规则；AI 基于这些规则生成一个版本；学生用具体例子检查实现是否忠实。若需要 AI 协助拟定需求，应让学生对每条建议说明接受、修改或拒绝的理由。AI 的作用是暴露可讨论的选择，不应替学生把所有选择一次性封闭。

5.1 用需求的可检验性连接语言与计算

需求表达可以按“对象—关系—规则—例外—验收”的顺序逐步展开。对象回答系统处理谁或什么；关系说明数据怎样关联；规则描述何时发生变化；例外明确不适用的条件；验收规定如何知道实现符合要求。它们是设计提示，不是需要机械填满的表格。简单任务可以只突出其中两项，复杂任务再逐步增加。

以校园植物观察记录为例，“记录植物长势”还不足以决定程序。学生需要确定记录对象是某株植物还是某个种类，日期能否重复，高度使用什么单位，缺失值如何显示。同一日期的两次测量是覆盖还是并存，会改变数据结构与统计结果。围绕这些具体选择进行讨论，比把提示改写得更像专业术语更接近计算建模。

AI 可以提出澄清问题，但问题的数量和难度应受控制。如果系统一次询问十余项工程细节，儿童可能选择全部接受或放弃自己的构想。教师可以要求模型每次只指出一个影响功能的歧义，再让学生用例子决定。学习证据体现在需求如何被澄清，以及学生是否理解选择的后果，而不在于提示文本是否越来越长。

5.2 保留需求变化的理由与可追溯关系

需求通常会随着运行结果变化。学生发现“提醒所有未归还者”可能暴露同伴信息，便把公开名单改为只显示自己的记录。这既涉及使用责任，也改变权限与输出规则。评

价时可以记录原要求、发现的问题、修改后的规则和验证方法，使需求修订成为可解释的学习过程。

对于语言表达仍在发展的学生，可先展示正常与异常的操作情境，再请其指出哪个结果符合想法。教师或辅助技术帮助转录，不应替学生决定规则。评价应关注选择的一致性和解释的计算意义，避免把书面表达流利度计入所有维度。这样既能保留自然语言入口的优势，也能减少语言能力对计算思维测量的干扰。

一个简洁的课堂记录可以只保留三项：本轮最重要的规则、一个能够检验规则的例子、一次有理由的修订。它比保存全部提示更便于师生讨论，也能够与后续测试形成对应关系。本文将其作为教学设计建议，尚不将这一特定记录方式视为经过独立验证的量表。

6 逻辑组织、问题分解与算法理解

自然语言生成容易隐藏程序结构。一个可以运行的作品可能由事件、状态、条件、循环和数据操作共同构成，但学生只看见界面变化。教学需要把部分内部结构重新显现出来，例如要求学生画出状态转换、预测条件分支或解释某一数据项的用途。无需每次阅读全部代码，也不能完全取消结构解释。

图 06 学生、机器人作品与共同项目（新加坡，2010 年）

注：VEX 机器人参赛团队历史照片；不提供个体计算思维成绩或 AI 辅助学习证据。 摄影或版权：Wrc Ynapmoc.；CC BY 3.0[32]；原始图片[33]。按原比例排版。

问题分解应按照任务依赖展开，而不是机械地要求“分成三个步骤”。在小游戏中，可以区分角色运动、碰撞判断、计分与结束条件，并说明模块如何交换信息。若计分在碰撞前更新和在碰撞后更新会产生不同结果，学生就有机会理解顺序与依赖，而不是只背诵分解的术语。

学生还应练习从具体作品中识别可以复用的规则。例如，借阅提醒和作业提醒都涉及对象、截止时间和状态变化，但它们对隐私、修改权限和重复通知的要求可能不同。抽象不是删除所有细节，而是保留当前问题需要的关系，同时说明什么条件下原有方案不能直接迁移。

AI 可以提供两种实现策略，要求学生比较数据流、复杂度和容易出错的部分。对年龄较小的学习者，比较可以通过卡片或角色扮演进行；对有基础的青少年，可以阅读关键函数或测试结果。重点是形成可解释的选择，而不是让学生从两个长答案中随意挑一个。

32 CC BY 3.0。原始资料

33 原始图片。原始资料

Hsu 提出的从编程到提示的建构主义设计框架强调有意义的任务、反思、迭代、社会参与和学习者主导。这属于概念框架，提供设计理由，并不是已经证实儿童长期计算思维提高的干预结果。本文据此把作品建构与反思相连接，同时保留独立评价。[34]

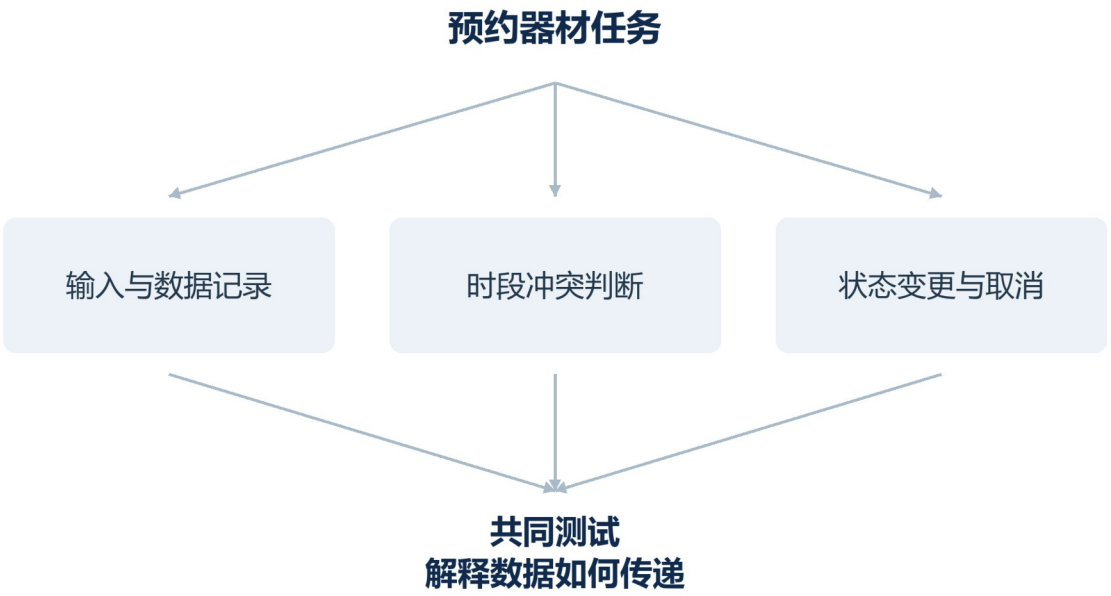

图 07 问题分解需要说明模块之间的依赖

注：本文虚构教学示例；仅使用模拟数据。

6.1 用表示转换暴露程序内部的因果关系

模块划分只有与输入、输出和依赖相连接，才具有学习意义。一个“计算得分”模块需要知道发生了什么事件、当前分数是多少，以及修改后的分数交给谁使用。学生可以用箭头说明数据流，再预测删除某个输入会发生什么。模型生成的函数名称可以作为线索，但名称本身并不能保证学生理解模块行为。

对于条件和循环，可以采用反事实提问：如果把“大于”改成“大于等于”，哪些输入的结果改变；如果计数器初值变为零，循环执行次数是否相同。此类问题将注意力集中在小而关键的结构上，避免要求初学者一次阅读完整应用。学生应先预测，再运行检验，最后解释差异；AI 可辅助解释，但不能替代预测环节。

6.2 将 PRIMM 作为可适配的阅读与创作路径

Sentance 等提出的 PRIMM 把预测、运行、探究、修改与创作组织为相互衔接的编程活动，为“先理解部分结构，再开展创作”提供了已有教学框架。它最初并非为生成

34 Hsu (2025)。文献[18]。原始资料

式 AI 设计；本报告将其作为适配参考，而不声称既有研究已经检验了这里提出的 AI 版本。[35]

在适配活动中，教师可提供一个规模受控的程序，让学生先预测运行结果，再提出不理解的位置；AI 只解释被指出的片段。随后学生修改一个条件或数据表示，检查行为怎样变化，最后把理解用于新的小任务。支持可以随学生表现逐渐减少，但不能依据完成时间快慢自动撤除，尤其需要区分熟练操作与真正理解。

图 08 PRIMM 框架的生成式 AI 教学适配路径

注：AISLI 依据 PRIMM 提出的教学适配示意；AI 版本尚需独立效果验证。[36]

这种路径不排斥自由创作。相反，短程序中的结构理解可以成为自主项目的准备。关键在于把创作阶段出现的新问题带回分析：某个功能为什么需要保存状态，两个事件同时发生时怎样处理，重用原模块需要改变哪些假设。只有当学生能在新问题中重新识别这些关系，才能谈论比单次运行更稳定的学习。

35 Sentance, Sue; Waite, Jane; Kallia, Maria (2019)。文献[19]。原始资料

36 Sentance, Sue; Waite, Jane; Kallia, Maria (2019)。文献[19]。原始资料

7 人机协作与元认知责任

有效协作需要学生知道 AI 正在承担什么工作。模型可以是解释者、代码起草者、测试建议者或错误定位助手，不应在同一次活动中不加区分地接管目标、实现和评价。一个系统自己生成代码、自己提出测试、再自己宣布通过，不能提供充分独立的验证。

本文提出分配与复核相连接的原则。学生把某项工作交给 AI 时，要同时说明预期输出和检查方法。例如，请 AI 解释报错时，学生应指出实际输入、期望输出与观察到的差异；请 AI 重构代码时，应保留关键测试以确认行为没有被意外改变。协作因此成为判断和证据组织的实践。

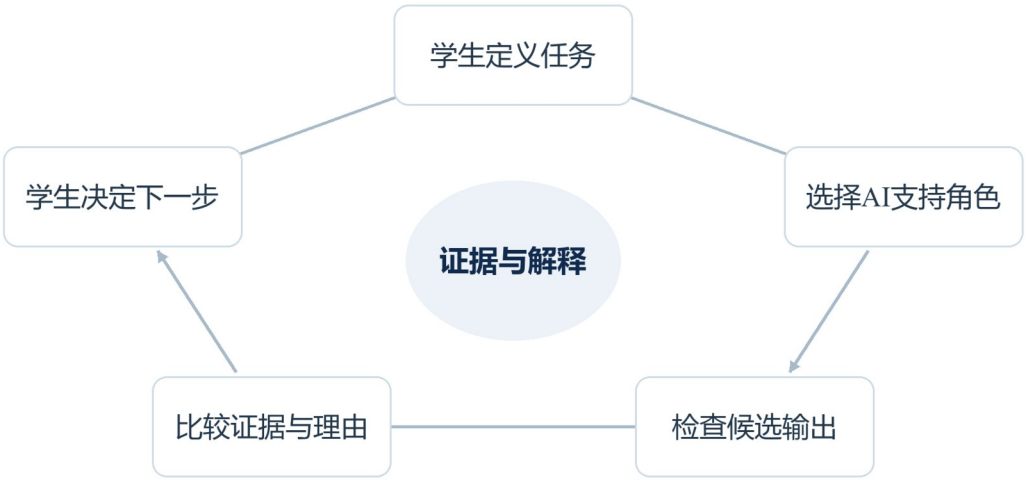

图 09 人机协作中的责任回路

注：本文设计框架；学生的选择与复核责任不能由模型自己代替。

元认知支持可以采用短而具体的停顿问题：我现在知道什么；我不确定哪一步；AI 的建议依据是什么；下一次测试可以区分哪两种解释。不要要求每轮对话都写长篇反思，以免形式负担压过学习。教师可以在关键决策点采集简短解释，结合版本记录判断学生是否真正参与。

同伴协作也不应被 AI 排除。两名学生可以轮换担任需求解释者和验证者，再共同决定是否接受模型输出。角色轮换有助于避免一人长期负责操作、一人只观看。对辅助沟通、阅读困难或不同语言背景的学生，分工应提供等价表达机会，不能只奖励打字速度快的人。

Why Johnny Can't Prompt 等人机交互研究表明，非 AI 专家设计提示会遇到实际困难。教育设计应通过清晰反馈和可理解的工具约束降低门槛，而非把“提示工程”包装成人

人都能立即掌握的简单技能。工具失败、界面含混和模型行为不稳定，应成为系统需要改进的部分。[37]

7.1 以共同任务为中心设计人机分工

人机分工不应只列出谁负责写代码，更需要明确谁设定目标、谁确认规则、谁判断测试是否充分。学生可以把语法实现交给 AI，但验收标准应在生成前讨论；也可以让 AI 提出两种算法，却需要由学生解释选择理由。责任分配应随教学目标变化，而不是固定为“人负责想法，机器负责一切细节”。

在小组学习中，还需要关注人与人之间的责任。设备操作者可能控制对话节奏，其他成员只看见最终输出。教师可以让一名学生口头提出预期，另一名学生记录测试，再轮换操作。共同解释一个失败案例通常比轮流点击更能显示参与质量。角色是帮助组织讨论的支架，不宜变成长期固定的能力标签。

7.2 用关键决策记录支持元认知

元认知监控应落在可以行动的问题上。例如，“我还不明白”可以进一步定位为“不知道变量何时改变”，从而决定下一步查看哪一段执行过程。AI 可以提示学生指出具体不确定之处，但如果自动补出所有诊断，学生就失去练习监控的机会。因此，系统宜先等待学生判断，再提供有限选择或解释。

表 05 人机协作中的责任与复核

关键决定	学生承担的判断	AI 可提供的支持
明确目标	选择规则并解释理由	提出有限澄清问题
实现方案	比较结构与依赖	生成片段或解释策略
诊断错误	提出假设与复现输入	定位线索和局部说明
验收作品	确认预期与已知限制	建议补充测试

注：AISLI 提出的教学设计建议；需结合年龄、任务和支持需要调整。

学习日志不必记录所有操作。可以选择目标改变、首次失败、重大修复和最终验收四类节点，要求学生简短说明当时相信什么、依据是什么、后来怎样修正。记录中允许

37 Zamfirescu-Pereira, J.D.; Wong, Richmond Y.; Hartmann, Bjoern; et al. (2023)。文献[20]。原始资料

“目前不知道”，并附下一步检查方法。这样可减少为了表现自信而接受模型输出的压力，也使教师更容易区分概念困难与工具问题。

当学生与 AI 意见不一致时，课程应支持通过运行、参考资料和同伴讨论解决，而不是把模型设为最终裁决者。教师同样可能需要重新检查自己的预期。把分歧转化为可检验的问题，有助于形成以证据修订观点的课堂文化；这种文化是人机协作学习的组织条件，不能仅靠增加一个“反思”按钮实现。

8 成果验证让直觉接受转向证据判断

成果验证是生成式 AI 辅助编程与计算思维之间最重要的连接之一。程序能够启动、界面看起来完整，只证明部分运行条件成立。学生需要区分功能正确、边界处理、数据可靠、使用安全和目标适切。对儿童而言，这些维度应通过具体任务逐步展开，而不是一次引入完整的软件工程清单。

测试可以从三个普通问题开始：正常输入会怎样；输入为空或不合要求会怎样；连续操作或改变顺序会怎样。随后加入极端值、重复操作和状态恢复。学生先预测结果，再运行，再解释差异，才能把测试从“点击看看”变成假设检验。AI 可以建议测试用例，学生仍需确认预期结果。

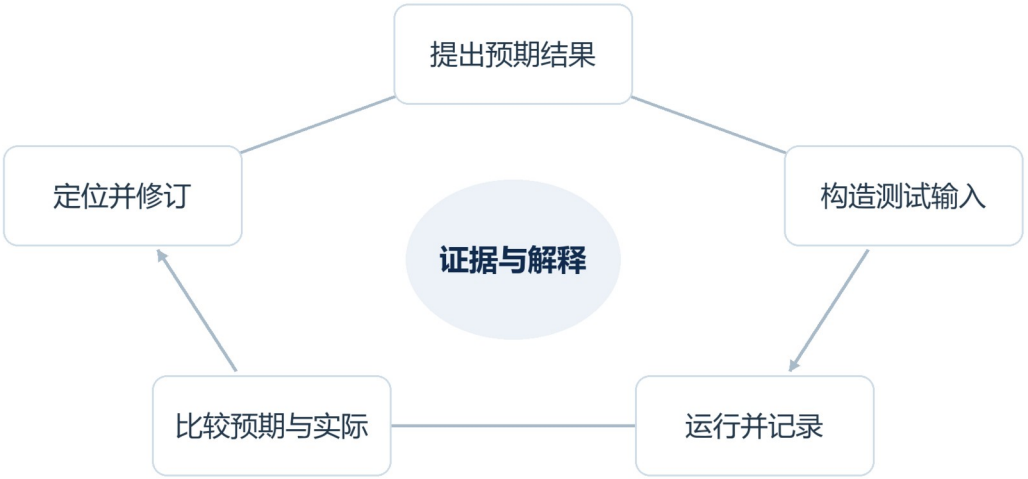

图 10 让测试成为假设检验

注：本文教学设计建议；模型提出的测试也需要核对预期。

以班级抽签程序为例，一次抽到合理姓名并不能证明规则正确。需要检查名单为空、重复姓名、抽完是否重置，以及是否符合是否放回的约定。若任务还声称公平，则需要

讨论随机过程与有限样本的关系。这个例子将条件、数据、随机性和验收连接起来，也暴露了“看起来没问题”的局限。

调试应围绕差异进行定位。先保存能复现问题的最小输入，再区分需求不清、实现错误、测试预期错误或运行环境问题。直接把整段项目交给 AI 反复“修好”，可能得到可用结果却不形成诊断能力。教师可以要求学生在每次重要修复前写一句假设，并在修复后指出什么证据支持该假设。

验证还包括来源与权限。学生可能在作品中使用个人数据、第三方图片或自动生成的外部依赖。课程应采用教师组织的适龄环境和最少数据，教学生解释哪些信息不需要上传、哪些资源需要许可，以及为什么不应将能运行等同于可以公开发布。这些内容服务于真实任务中的责任判断，不代替基本计算概念教学。

8.1 使测试能够发现错误，而不只是确认期待

测试首先需要独立的预期结果。如果代码和测试答案都由同一轮模型输出提供，二者可能共享同一个误解。学生可以先手工计算一个简单情形，用规则卡片描述边界，再让 AI 建议其他用例。所谓独立不是完全拒绝工具，而是避免让被检验的实现同时决定全部正确标准。

在班级抽签任务中，学生可先约定不放回规则，并构造只有两人的小名单。连续抽取三次时，第三次应出现什么，是可以在生成前确定的结果。随后再扩大名单，检查重置和重复姓名等情况。小规模输入能够让学生看清状态变化，比一开始使用复杂真实数据更适合建立验证习惯。

图 11 从可运行作品到有依据的验收

注：AISLI 综合设计的教学框架；非实验数据。

8.2 将错误分类转化为有效调试行动

错误至少可能来自要求、实现、测试预期和环境。要求互相冲突时，继续修改代码未必有用；测试预期写错时，一个正确程序也会被判失败；依赖缺失时，算法解释不能修复运行环境。学生先尝试判断错误位于哪一层，再请求对应帮助，能够使对话更有方向，也减少无依据的反复生成。

教师可以提供一段带有明确缺陷的短程序，让学生设计能够暴露缺陷的输入。这类任务把“发现问题”作为成功，而不只奖励修复后可运行。评价可以观察学生是否缩小输入范围、保留复现步骤，以及修复后是否回查原有功能。测试数量不是唯一指标，少量具有区分力的测试可能比大量相似点击更有价值。

成果验证还需要适度。小学入门课程无需承担专业系统审计，但可以坚持“说明预期—检验关键情形—记录已知限制”。对较复杂的青少年项目，再逐步增加权限、数据来源和依赖管理。没有检查过的功能应标为未检查，而不是借助模型生成一句“全部通过”。这种如实报告限制的能力，应当成为作品展示的一部分。

9 年龄、先备知识与包容性设计

年龄只是设计线索，不能替代对学习者已有知识、读写能力、执行功能与支持需要的了解。相同年龄的学生可能具有很不同的编程经验。课程应从短任务了解学生能否解释事件与条件、预测输出和表达约束，再决定模型可以提供多大帮助。

图 12 Ghana Code Club 成员的数字工作坊参与（2018 年）

注：真实维基百科工作坊；用于呈现参与机会，不对应儿童 vibe coding 干预。 摄影或版权：Mwintirew；CC BY-SA 4.0[38]；原始图片[39]。按原比例排版。

对于初次接触计算活动的儿童，可以使用实物、图块和口头描述表示规则，由教师控制生成环境。学生负责预测与比较，小范围改变一个条件，再观察结果。对于具有图形化编程经验的学生，可以把图块结构与 AI 生成的实现对应起来，防止界面升级掩盖概念断层。

对于具备一定文本编程经验的青少年，可以加入模块接口、数据结构、测试覆盖和版本差异，逐步要求阅读关键代码与解释设计选择。增加复杂度的依据应是学生能够承担新的判断，而不是作品能够被模型生成得更庞大。必要时降低作品规模，换取更深的理解和更可靠的验证。

38 CC BY-SA 4.0。原始资料

39 原始图片。原始资料

表 06 根据先备知识调节支持的设计建议

学习基础	学生应承担的活动	可提供的 AI 帮助
初次接触计算	预测规则结果与解释差异	教师控制的有限生成
具有图形化经验	对照结构与测试条件	解释代码与生成局部
具备文本编程经验	模块接口与关键代码解释	实现备选和调试线索
需要不同表达支持	以口述图示操作说明规则	转换表达并保持学生选择

注：本文建议；不能仅按年龄机械分组。

自然语言入口具有包容潜力，也可能制造新的不平等。擅长书面表达、拥有更多设备时间或付费模型的学生可能得到更好结果。课堂应提供口述、图示、范例比较与共享设备轮换，并评价学生对规则的理解。使用辅助技术不应被视为较低独立性，关键是学生是否能够表达、选择和验证自己的方案。

AI 素养框架强调以人为本、伦理、技术应用和系统设计等能力。儿童课程可据此讨论模型怎样产生输出、何时可能错误，以及谁应对决定负责。学习者有权质疑和不用某个建议；这种能动性应通过实际任务练习，而不是只记住几条安全口号。[40]

9.1 从年龄分层转向支持需要的识别

课程可以用几个低负担任务了解起点：学生能否预测一条条件规则，能否根据例子解释变量，能否发现重复记录造成的问题。起点观察服务于提供帮助，不用于给儿童贴上固定的能力标签。学生在语言表达、视觉表示与实际操作中的表现可能不同，支持设计应允许优势通道参与。

当模型解释包含大量抽象术语时，教师可以要求将其对应到当前作品中的对象与行为，而不是不断增加文字长度。对于阅读支持需要较高的学生，短句、语音、图示或分步呈现可能更合适；具体方案仍应根据学习者体验调整。可访问性支持应与学习目标兼容，不能因为使用语音或辅助输入就降低对规则理解的期待。

9.2 避免工具差异成为隐蔽的课程门槛

如果某些学生在课外拥有更强模型、更多调用额度或家庭技术支持，他们可能带来更复杂的作品。课堂评价应确认哪些部分在共同条件下完成，并提供校内可获得的资源。

40 UNESCO (2024)。文献[21]。原始资料

作品复杂度可以作为创作特征，但不能直接解释为个人能力差异。更公平的评价会包含短时、受支持且条件明确的现场解释与修改任务。

教师还需考虑不同语言背景。模型对日常表达、方言或儿童不完整句子的响应可能不同。学生获得不合适输出时，问题可能来自系统对表达方式的适应不足，而非需求思考本身不足。让学生先向同伴说明规则，再比较模型如何理解，有助于区分沟通意图与机器处理之间的差异。

适龄设计也包括节奏与退出选择。有些学生可能希望先使用图块或手工编程理解问题，再尝试 AI；这不应被视为落后。课程可以提供通向相同概念目标的不同活动路径，并通过共同的预测、解释和验证任务评价。包容性由学习机会和参与质量体现，不由所有人是否使用同一界面体现。

10 评价作品、过程与独立能力

作品评价应检查是否满足任务要求，过程评价应观察学生怎样表示问题、选择策略和验证，独立评价则检验支持撤除后能够保留什么。三者相互补充。只看最终作品容易奖励模型能力与工具资源；只看过程日志又可能奖励善于记录的学生；仅用一次无工具测验则可能遗漏真实协作能力。

图 13 作品过程与独立能力的联合评价

注：本文评价框架；三类证据分别报告，避免作品分替代学习分。

本文提出一套待验证的课堂量规，覆盖需求表述、结构分解、算法解释、人机分工、测试调试和迁移六个维度。每项要求对应可观察证据，例如能否指出一个边界情况、说

明模块之间的数据传递、拒绝一个没有依据的 AI 修改。它是设计建议，不是经过常模化的心理测量工具，不宜直接用于跨学校排名。

表 07 计算思维过程的课堂观察量规

维度	初步形成	可独立解释
需求表述	说明目标和一般功能	明确输入输出约束与边界
问题分解	列出部分步骤	说明模块关系与依赖
算法理解	描述程序做了什么	预测条件变化后的结果
人机协作	能够提出请求	分配任务并说明采纳理由
测试调试	运行正常用例	构造边界并定位差异
迁移	复现相似作品	解释可复用规则与限制

注：本文提出的教学观察工具；未进行信效度与常模验证。

过程数据不应只计数。更多提示可能说明深入迭代，也可能说明目标混乱；较少提示可能说明已有规划，也可能说明直接接受输出。应结合对话内容、版本变化、测试与学生解释识别活动性质。研究编码应报告定义、示例和一致性，并把 AI 自己撰写的解释与学生独立解释区分。

独立评价可以使用结构相似、表面不同的任务。例如，学生在 AI 支持下完成借阅提醒后，独立解释一个器材预约系统需要哪些状态和冲突规则。若课程目标是调试，可以提供一个小错误程序，要求定位与说明，而不必让所有学生从零编写完整应用。

保留与迁移需要时间。即时后测可以说明课程结束时的表现，延迟测验与新任务能够检验是否形成可持续能力。研究应分别报告有 AI 和无 AI 条件、即时和延迟结果，并关注不同先备水平的变化。没有这些数据时，应明确报告未测量，而不是以学生喜爱程度填补。

10.1 建立作品、过程、保持和迁移的评价组合

作品评价关注是否实现约定功能，过程评价关注规则如何形成与修订，保持评价关注一段时间后能否重现理解，迁移评价关注能否在不同表面情境中运用关系。它们回答不同问题，不能通过一个总分掩盖差异。学生可能作品优秀但解释薄弱，也可能作品简朴而判断清楚；反馈应指出具体学习位置。

独立评价不一定要求完全徒手编写长程序。可以让学生预测输出、补全流程、修改关键条件或挑选能够发现错误的测试。评价条件要明确哪些工具允许使用、哪些支持已撤除。对依赖辅助沟通或其他可访问性工具的学生，应区分必要支持与待评价的 AI 帮助，避免以撤除全部技术作为“独立”的唯一含义。

表 08 计算思维评价的互补任务

评价层次	任务示例	避免的误判
即时作品	检查功能与约定规则	复杂作品不等于独立理解
过程判断	解释一次有理由的修订	对话次数不等于思考质量
保持	后续时点预测与修改	即时成绩不等于长期掌握
迁移	用相同关系处理新情境	表面换题不一定构成迁移

注：AISLI 综合提出的评价设计；不是经验证的标准化量表。

10.2 提高量规的可解释性与评分一致性

量规中的“逻辑清晰”需要转化为可观察描述。例如，需求中的条件是否一致，流程是否涵盖关键分支，测试是否对应验收要求，解释能否说明状态变化。每一维度宜配正例、边界例与常见误判，帮助教师在同一标准下讨论。若量规由本文提出，应明确它是设计草案，需要后续试用与验证。

评分时可以先隐藏学生使用的工具品牌，减少对先进工具或复杂外观的印象影响。对于开放作品，教师之间可对部分样本进行独立评分，讨论分歧并修订描述。对话日志可作为补充，但不应成为全天候监控；记录范围应与评价问题相称，允许学生解释日志之外发生的讨论与思考。

研究报告还应说明测量误差与缺失。学生缺席延迟测验，不能直接按零分处理；只有完成所有任务者进入分析，可能高估效果。若不同年龄使用不同难度题目，也不能把原始分数直接排序。透明报告这些条件，比给出一个看似精确的“计算思维提升百分比”更能支持解释和复核。

11 研究设计与可检验假设

后续研究应把工具与教学设计拆开。可以比较无 AI 的优质编程教学、可直接生成完整代码的 AI 环境，以及要求计划、解释和验证的 AI 环境。课时、任务难度、教师支

持和评价条件应尽量一致。否则，观察到的差异可能来自额外教学时间或教师关注，而不是 AI 交互本身。

第一项可检验假设是，结构化需求表达活动可能通过改善问题表示，帮助学生完成陌生任务。研究应同时测量表示质量与独立问题求解，不能只用作品得分推定机制成立。第二项假设是，要求先预测再测试可能促进调试与逻辑理解，但可能增加初期负担；需要考察不同先备水平与任务复杂度。

第三项假设涉及协作责任。让学生明确选择 AI 角色并说明接受理由，可能减少不加判断的采纳。然而，简单增加反思表格也可能导致形式化填写。研究应检查真实对话、版本和解释之间是否一致，并收集学生对认知负担和自主感的反馈。

表 09 后续研究中的关键变量与测量

研究问题	需要比较的条件	主要证据
需求表达是否促进学习	结构化表述与一般提示	表示质量与独立迁移
验证是否改善调试	先预测再测试与直接运行	错误定位与解释
协作责任是否抑制盲从	明确分工与自由代理	采纳理由与错误拒绝
效果能否保持	有工具与无工具延迟任务	保持及跨情境表现
是否扩大差距	不同先备语言和支持条件	分组结果与退出原因

注：本文研究设计建议；尚无统一效应结论。

一般教育中的生成式 AI 研究提醒我们，有工具时表现改善与撤除工具后学习可能分离。Bastani 等数学教育研究提供这一邻近证据，但它不是编程或计算思维研究。本文用它说明为什么需要无工具测量，不将其效果数字迁移到儿童 vibe coding 课程。[41]

研究报告应记录模型版本、提示与系统约束、代码可见程度、自动执行权限、教师干预、工具故障、退出情况和不利事件。班级或教师层面的组织需要相应统计处理；多种结果应事先确定主要指标，并报告未显著和不利结果。短期试点可以提供实施线索，长期发展结论需要更长的证据链。

41 Bastani et al. (2025)。文献[22]。原始资料

11.1 把教学机制转化为可以比较的研究条件

较有价值的比较不是只有“有 AI”和“无 AI”，还可以考察先预测后生成与直接生成、解释支架与完整答案、固定提示与根据困难调整的提示。各条件应尽量保持任务、时间、教师关注和设备机会相近。否则，观察到的差异可能来自教学投入不均，而不是所研究的支架机制。

研究可以预先指定一个主要结果，例如撤除生成帮助后的规则迁移，再把任务完成、兴趣和认知负荷作为补充结果。这样能够降低只挑选显著结果进行叙述的空间。样本规模应依据预期差异、测量可靠性和班级聚集程度规划，不宜根据已有几篇小样本研究随意设定统一最低人数。

11.2 将时间维度与情境差异纳入设计

儿童使用工具的早期兴奋可能影响投入，随着熟悉度上升，使用策略也可能改变。因此，应区分入门阶段、相对稳定使用和支持逐步减少后的表现。重复使用同一道测验可能带来练习效应；完全不同的测验又可能难度不等。平行任务、明确的评分依据和适当的延迟安排，可以帮助解释时间变化。

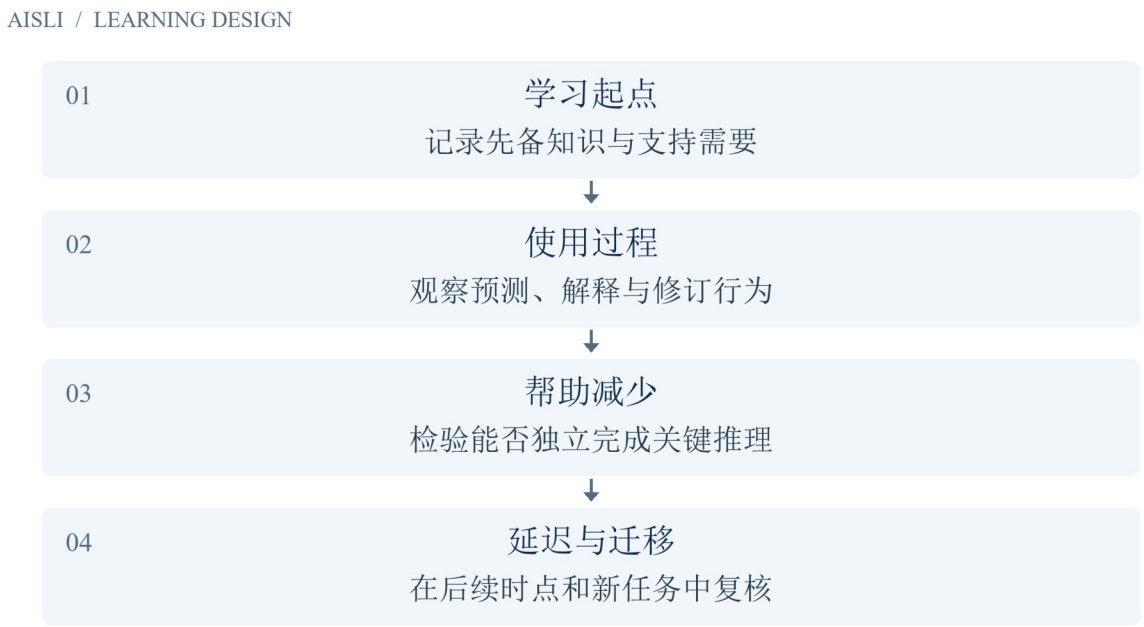

图 14 计算思维学习证据的时间与迁移路径

注：AISLI 综合提出的研究设计示意；各阶段的间隔与任务需按研究问题确定。

多校研究能够增加情境多样性，但学校数量和学生数量都需要报告。若干班级的数千条对话不是数千名独立参与者。分析应考虑学生、班级和任务之间的嵌套关系，并说

明结论是在个人层面还是班级层面成立。模型升级、提示修改和网络中断也可能改变干预，宜作为实施信息记录。

质性材料可以解释为何相同支架产生不同体验，但访谈对象的选择应透明。只访问高分学生会遗漏挫败与退出；只展示成功作品会掩盖教学成本。把不同表现、不同支持需要与未完成案例纳入分析，能够提出更可信的机制解释。后续开放材料应保护未成年人，优先共享任务、量规、去标识的结构化结果与分析代码。

12 面向课程与工具的综合建议

课程设计应围绕需要学生承担的计算活动展开。选择一个有意义而规模可控的问题，使学生先表示目标与规则，再与 AI 协作实现，通过预测、测试、解释和迁移结束任务。图形化、自然语言和代码可以共同服务这个过程，不必把某一种表达方式宣布为唯一未来。

图 15 社区编程工作坊中的学习者与支持者（2016 年）

注：Black Boys Code 活动合影；用于呈现社区支持，不作为 AI 学习效果证据。 摄影或版权：Bryan R. Johnson / Black Boys Code；CC BY-SA 4.0[42]；原始图片[43]。按原比例排版。

42 CC BY-SA 4.0。原始资料

43 原始图片。原始资料

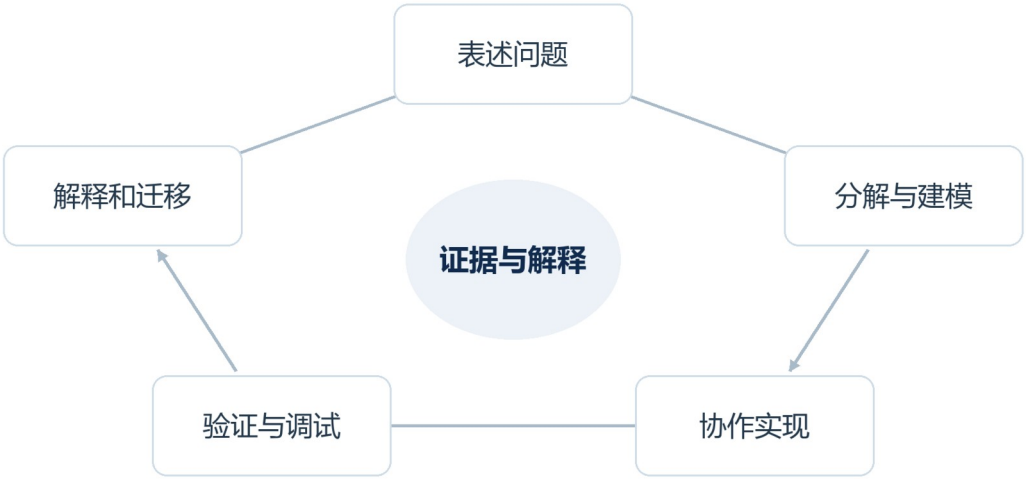

图 16 面向计算思维的 AI 协作学习循环

注：本文综合设计建议；尚未作为统一教学模型验证。

工具应支持分级帮助和证据保存：先提示再解释，允许显示关键结构，保留版本差异，帮助学生形成测试，但不自动宣布全部合格。学生应能回看自己提出的要求、模型改变的内容和检验结果。对于初学者，少量清楚的选择通常比一次展示完整代理工作流更容易理解。

教师发展应包括分析真实学生过程，而不只学习最新平台操作。教师需要辨别哪些困难值得保留、哪些摩擦可以消除，以及什么证据说明学生正在理解。可以共同审阅同一份需求稿、AI 对话和测试记录，讨论不同支持方式可能改变什么学习活动。

学校层面应提供公平的工具条件、适龄治理和明确的评价规则，不以是否使用 AI 简单区分诚实与否。对允许使用的部分，应要求透明披露和解释；对独立能力评价，应明确工具边界并保证任务适切。评价重构应与教学同步，否则学生会被鼓励在平时依赖自动生成，又在评价时突然失去全部支持。

综合判断是：生成式 AI 辅助编程为计算学习提供了新的入口，但发展效应取决于学习责任怎样分配。需求表达只有转化为可检查的模型，逻辑组织只有连接到执行与解释，人机协作只有保留选择和复核，成果验证只有以证据为标准，才可能共同促进计算思维。现阶段最有价值的主张是建立这些条件并严谨检验，而不是提前宣布它必然带来能力革命。

12.1 课程实施的递进路径

学校可以从范围明确的小任务开始，先确定一个计算概念与一个验证行为，再选择生成式 AI 如何提供帮助。试用阶段应观察学生是否理解规则、教师是否能够识别困难，

以及工具解释是否与课程一致。只有在这些条件较稳定时，才逐步增加作品规模和自主程度。扩展依据应是学习证据，而不是一次展示活动的视觉效果。

教师发展需要同时覆盖计算概念、反馈设计与工具判断。仅学习提示模板容易使培训停留在操作层面；仅强调风险又难以帮助教师组织实际活动。可围绕同一短任务比较不同支架，讨论哪些工作应由学生承担，再分析学生的预测与测试记录。这样培训产物可以直接进入课堂，并接受后续修订。

12.2 将工具设计与教育评价连接起来

适合学习的编程助手应支持查看版本差异、保留学生原有规则、按需要解释局部代码，并允许教师设置帮助范围。所谓帮助范围不宜只通过禁止关键词实现，因为学生真实困难可能被误拦。更有价值的是清楚显示系统当前提供哪类支持，并使学生能够回到自己的问题和证据。

报告采用“生成式人工智能辅助编程”这一学术名称，正是为了将讨论从某一种流行操作风格扩展到可比较的学习活动。vibe coding 仍是其中的重要问题情境：它提醒研究者关注快速生成与理解责任之间的张力。但研究结论应依据实际教学设计，而不是依据名称的新颖程度。

未来最需要积累的，是儿童在持续使用、支架调整和跨任务迁移中的发展证据。研究者可以公开失败案例和边界条件，学校可以记录教师工作量与可访问性体验，工具开发者可以报告模型变更与支持逻辑。各方共同形成可复核的证据，才能判断哪些帮助真正增加了学习者表达、组织、协作与验证的能力。

附录 A 一个完整学习任务的设计示例

任务为“为班级公共器材设计预约原型”，只使用虚构姓名和设备，不连接真实个人信息。教师给出基本需求：选择器材与时段，避免重复占用，允许取消，并显示当前状态。学生补充约束，画出数据与状态，说明同一时段冲突如何处理。任务用于学习表示、条件、数据与测试，不要求建立真实运行平台。

第一次生成之前，学生提交简短需求稿与三个预期测试。生成之后，先检查正常预约，再检查重复预约、取消后重约和空输入。发现问题时保存最小复现步骤，提出原因假设，再向 AI 请求有限修改。每轮只改变可理解的一部分，使版本比较与解释仍然可行。

小组内轮换需求解释者、实现观察者和测试者；人数不足时合并角色。结束时选择一条 AI 建议说明为何接受或拒绝，并解释一个关键状态变化。教师用学生自己的语言追问，不要求重复模型术语。评价同时看原型是否满足要求、测试是否有效和解释是否一致。

迁移任务改为“实验室材料领取登记”，学生不使用 AI，指出哪些规则可以复用、哪些必须改变。这个任务提供教学设计示例，不代表已经验证的最佳课程，也不预设固定课时或统一年龄。教师应依据学生先备知识和可用支持调整。

附录 B 研究透明性与更新重点

本工作版本以 AISLI 为署名，由 AI 完成定向资料检索、可获得来源比对、综合写作、原创图表和文档排版，尚未经过独立同行评议或正式出版审定。本文的循环模型、课堂量规与课程示例均为综合提出的设计建议，不是已经验证的量表或统一教学方案。

后续更新优先补齐三类证据：儿童 vibe coding 研究的原始全文与完整方法；持续一个学期以上、包含延迟和迁移评价的研究；适用于不同语言、先备知识和支持需要的公平分析。引用的成人研究与概念框架继续保留边界，不随研究数量增加而自动升级为儿童发展因果证据。

参考文献

按首次引用顺序编号；题录或摘要级可得性限制在正文中说明。网络资料核验截止 2026 年 9 月 11 日。

[1] Wing (2006). Computational Thinking. 原始来源

[2] Brennan & Resnick (2012). New frameworks for studying and assessing the development of computational thinking. AERA 2012. 原始来源

[3] Grover, Shuchi; Pea, Roy (2013). Computational Thinking in K–12. Educational Researcher. 原始来源 DOI: 10.3102/0013189x12463051

[4] Weintrop et al. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. 原始来源

[5] Risko, Evan F.; Gilbert, Sam J. (2016). Cognitive Offloading. Trends in Cognitive Sciences, 20(9), 676–688. 原始来源 DOI: 10.1016/j.tics.2016.07.002

[6] Kazemitabaar, Majeed; Chow, Justin; Ma, Carl Ka To; et al. (2023). Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. Proceedings of CHI 2023. 原始来源 DOI: 10.1145/3544548.3580919

[7] Yan, Yi-Miao; Chen, Chuang-Qi; Hu, Yang-Bang; et al. (2025). LLM-based collaborative programming: impact on students' computational thinking and self-efficacy. Humanities and Social Sciences Communications. 原始来源 DOI: 10.1057/s41599-025-04471-1

[8] Si, Janice Jianing; Li, Donglin; Wang, Qiuning; et al. (2026). Exploring the Impacts and Challenges of V ibe Coding Paradigm to Children's Programming Learning and Practices. Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems. 原始来源 DOI: 10.1145/3772318.3790366

[9] Xu, Jie; Sun, Dan; Feng, Yue; Shadiev, Rustam; Li, Yan (2026). Empirical insights into the influence of ChatGPT-facilitated programming on primary learners' performances: a mixed methods approach. International Journal of STEM Education, 13, 18. 原始来源 DOI: 10.1186/s40594-026-00606-1

[10] Guo, R.; Li, G.; Miao, H.; Pi, Z.; Xie, L. (2026). Impact of Generative AI-assisted programming on the computational thinking of high school students. Frontiers in Psychology, 17, 1703177. 原始来源 DOI: 10.3389/fpsyg.2026.1703177

[11] Liu, Jinfang; Zhang, Yi; Li, Wei; Wang, Qiyun; Niu, Pingxiu; Zhang, Xue (2026). Adaptive vs. planned metacognitive scaffolding for computational thinking: Evidence from generative AI-supported programming in elementary education. Computers & Education, 241, 105473. 原始来源 DOI: 10.1016/j.compedu.2025.105473

[12] Thorgeirsson, Sverrir; Weidmann, Theo B.; Su, Zhendong (2026). Computer Science Achievement and Writing Skills Predict Vibe Coding Proficiency. Proceedings of CHI 2026；公开作者稿 arXiv:2603.14133. 原始来源 DOI: 10.1145/3772318.3791666

[13] Prather, James; Reeves, Brent; Leinonen, Juho; et al. (2024). The Widening Gap: The Benefits and Harms of Generative AI for Novice Programmers. arXiv:2405.17739（所引版本为公开预印本）. 原始来源

[14] Kazemitabaar, Majeed; Ye, Runlong; Wang, Xiaoning; et al. (2024). CodeAid: Evaluating a Classroom Deployment of an LLM-based Programming Assistant that Balances Student and Educator Needs. arXiv:2401.11314（所引版本为公开预印本）. 原始来源

[15] Nguyen, Sydney; Babe, Hannah McLean; Zi, Yangtian; et al. (2024). How Beginning Programmers and Code LLMs (Mis)read Each Other. arXiv:2401.15232（所引版本为公开预印本）. 原始来源 DOI: 10.1145/3613904.3642706

[16] Gama, Kiev; Calegario, Filipe; Jackson, Victoria; et al. (2025). "Can you feel the vibes?": An exploration of novice programmer engagement with vibe coding. arXiv:2512.02750（所引版本为公开预印本）. 原始来源

[17] Pimenova, Veronica; Fakhoury, Sarah; Bird, Christian; et al. (2025). Good Vibrations? A Qualitative Study of Co-Creation, Communication, Flow, and Trust in Vibe Coding. arXiv:2509.12491（所引版本为公开预印本）. 原始来源

[18] Hsu (2025). From Programming to Prompting Developing Computational Thinking through Large Language Model-Based Generative Artificial Intelligence. 原始来源

[19] Sentance, Sue; Waite, Jane; Kallia, Maria (2019). Teaching computer programming with PRIMM: a sociocultural perspective. Computer Science Education, 29(2–3), 136–176. 原始来源 DOI: 10.1080/08993408.2019.1608781

[20] Zamfirescu-Pereira, J.D.; Wong, Richmond Y.; Hartmann, Bjoern; et al. (2023). Why Johnny Can’t Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. 原始来源 DOI: 10.1145/3544548.3581388

[21] UNESCO (2024). AI competency framework for students. 原始来源

[22] Bastani et al. (2025). Generative AI without guardrails can harm learning. 原始来源

AISLI

让创造连接理解与判断

研究综述 / 2026 年 9 月

中文工作版本