

少儿编程教育的价值重构与实践路径研究

张金钰

库车新梦想信息技术有限公司 新疆 库车 842000

摘要: 本文通过分析全球少儿编程教育发展现状,结合认知发展理论与建构主义教育观,探讨编程教育对儿童思维发展的促进作用。研究采用案例分析法与实证研究法,揭示当前少儿编程教育面临的挑战,并提出基于核心素养培养的实践路径。研究发现,编程教育在提升逻辑思维、培养创新能力、塑造数字公民意识等方面具有显著价值。

关键词: 少儿编程教育; scratch; Python

1 绪论

1.1 研究背景

在当今数字化时代,数字经济的蓬勃发展正以前所未有的速度重塑着社会的各个领域,对人才能力结构也提出了全新的要求。在这样的大背景下,编程能力作为数字化技能的核心组成部分,逐渐成为了教育领域关注的焦点。

从全球范围来看,各国纷纷出台编程教育相关政策,大力推动编程教育的发展。英国在2014年率先将编程纳入K12必修课程,旨在让学生从小接触编程,培养计算思维和数字化技能,为未来的职业发展和社会生活做好准备。美国则发起了“计算机科学为所有”国家行动计划,投入大量资源,致力于让每个学生都有机会学习计算机科学和编程,提升全民的数字素养。在中国,随着《新一代人工智能发展规划》的出台,编程教育也得到了政策层面的大力支持,各地学校纷纷开展编程课程,编程培训机构如雨后春笋般涌现,社会对编程教育的重视程度不断提高。

教育技术的不断进步也为少儿编程教育的普及提供了有力支持。Scratch、Python、机器人编程等编程工具的出现,使得编程学习变得更加生动有趣、易于上手。Scratch以其图形化的编程界面,让孩子们通过拖拽积木块的方式就能编写程序,极大地降低了编程的门槛,激发了孩子们的学习兴趣。Python作为一种简洁高效的编程语言,不仅在人工智能、数据分析等领域广泛应用,也逐渐成为少儿编程教育的重要工具,培养孩子们的逻辑思维和代码编写能力。机器人编程则将编程与硬件相结合,让孩子们通过编写程序控制机器人的行动,实现创意和想法,进一步提升了孩子们的动手能力和实践能力。

1.2 研究意义

从理论层面来看,本研究有助于拓展STEAM教育

理论边界。STEAM教育强调科学(Science)、技术(Technology)、工程(Engineering)、艺术(Art)和数学(Mathematics)的跨学科融合,而少儿编程教育作为STEAM教育的重要组成部分,不仅涉及到技术和数学领域,还与科学、工程以及艺术等学科有着紧密的联系。通过深入研究少儿编程教育的价值和实践路径,可以进一步丰富和完善STEAM教育理论体系,为跨学科教育的发展提供理论支持。

在实践方面,本研究的成果将为教育决策者提供重要的实施参考。当前,少儿编程教育在全球范围内迅速发展,但在实施过程中也面临着诸多问题,如课程体系不完善、师资力量不足、评价体系不健全等。本研究通过对全球实践模式的分析和现实困境的探讨,提出了基于核心素养培养的实践路径和解决方案,为教育决策者制定相关政策和规划提供了科学依据,有助于推动少儿编程教育的健康、有序发展。

2 少儿编程的教育价值

2.1 认知发展视角

皮亚杰的认知发展阶段理论为理解少儿编程教育的认知价值提供了重要框架。在具体运算阶段(7-11岁),儿童开始具备一定的逻辑思维能力,但仍需要具体事物的支持。少儿编程中的图形化编程工具,如Scratch,通过直观的图形界面和简单的操作方式,帮助儿童将现实世界中的问题和现象转化为符号化的编程指令,从而培养他们的符号化思维能力,对抽象概念的理解和运用能力。

当儿童进入形式运算阶段(11岁以上),他们的思维开始摆脱具体事物的束缚,具备了抽象逻辑思维能力。此时,引入Python等文本编程语言进行编程教育,可以进一步训练他们的抽象逻辑思维。在Python编程中,儿童需要使用变量、函数、循环等抽象概念来构建程序,解决复杂的问题。通过不断地编写代码、调试程序,他

们的逻辑思维能力得到了锻炼和提升,学会了运用抽象的逻辑规则来分析和解决问题。

计算思维作为编程教育的核心目标之一,其培养路径贯穿于编程学习的全过程。问题分解是计算思维的基础,通过将复杂的问题分解为一系列简单的子问题,儿童学会了有条理地分析问题,将大问题逐步细化为可解决的小问题。

抽象建模是将现实世界中的问题或现象抽象为数学模型或程序模型的过程,它帮助儿童理解问题的本质,并用计算机语言进行表达。在编程教育中,儿童可以通过编写程序来模拟物理现象、经济模型等,将抽象的概念具象化,加深对知识的理解和应用。

算法设计是计算思维的高级体现,它要求儿童设计出解决问题的具体步骤和方法,并将其转化为可执行的代码。通过学习算法设计,儿童不仅提高了编程能力,还培养了创新思维和解决实际问题的能力。

2.2 核心素养培育

在信息时代,信息素养已成为公民必备的基本素养之一。少儿编程教育可以有效提升儿童的数据获取与处理能力。在编程过程中,儿童需要从各种数据源获取信息,如传感器数据、网络数据等,并对这些数据进行分析和可视化处理。通过这些实践活动,他们学会了如何运用编程工具来收集、整理和分析数据,从而更好地理解 and 利用信息,提升了信息素养。

工程思维强调运用系统化的方法来解决实际问题,包括需求分析、设计规划、实施验证等环节。在少儿编程教育中,通过开展项目式学习,儿童可以亲身体验工程思维的全过程。例如,在设计一个机器人项目时,儿童需要首先明确项目的需求和目标,然后进行设计规划,选择合适的硬件和软件平台,编写程序实现机器人的功能,最后对项目进行测试和优化。在这个过程中,他们学会了如何运用工程思维来解决实际问题,培养了系统分析和解决问题的能力。

创新意识是推动社会进步的重要动力,少儿编程教育为培养儿童的创新意识提供了广阔的空间。在编程学习中,儿童可以通过不断地尝试和迭代开发,将自己的创意转化为实际的作品。例如,在Scratch编程中,儿童可以自由地设计游戏、动画和故事等,发挥自己的想象力和创造力。同时,通过与其他同学的交流和分享,他们可以获取更多的灵感和思路,进一步激发创新思维,培养成长型思维模式,即相信通过不断努力和学习的,自己的能力可以得到提升^[1]。

2.3 社会价值延伸

在数字化时代,儿童作为数字原住民,从小接触各种数字技术和信息。通过编程学习,儿童可以了解到数字技术的原理和应用,培养对数字技术的敬畏之心和责任感,成为有担当的数字公民。

随着人工智能技术的快速发展,人工智能伦理问题日益受到关注。少儿编程教育可以作为人工智能时代的伦理启蒙,引导儿童思考人工智能的发展对社会和人类的影响。在编程教学中,可以引入一些与人工智能伦理相关的案例和讨论,让儿童了解人工智能可能带来的风险和挑战,培养他们的伦理意识和价值观,使他们在未来能够正确地应用人工智能技术。

少儿编程教育具有很强的跨学科性,它融合了数学、科学、技术、艺术等多个学科的知识 and 技能。通过编程学习,儿童可以将不同学科的知识进行整合和应用,打破学科界限,培养跨学科知识整合能力。例如,在设计一个编程作品时,儿童可能需要运用数学知识来计算坐标和角度,运用科学知识来模拟物理现象,运用艺术知识来设计界面和角色形象,从而实现多学科知识的有机融合。

3 实践模式分析

PBL项目式学习(Project - Based Learning)以真实的项目为驱动,让学生在完成项目的过程中学习和应用知识。在少儿编程教育中,以机器人竞赛为例,学生需要组成团队,根据竞赛主题和要求,设计、搭建和编程控制机器人,完成一系列任务。在这个过程中,学生不仅学习了编程知识和机器人技术,还培养了团队协作能力、问题解决能力和创新能力。

双师课堂模式是指学校教师与工程师协作进行教学。学校教师负责基础知识的讲解和课堂管理,工程师则凭借其丰富的实践经验,为学生提供实际项目案例和技术指导。这种模式能够充分发挥学校教师和工程师的优势,让学生在在学习理论知识的同时,了解行业的最新动态和实际应用,提高学生的学习兴趣和学习效果。

硬件编程平台也在不断更新换代,从Micro:bit到Arduino再到树莓派,它们的功能越来越强大,应用场景也越来越广泛。AI模型工具的出现为少儿编程教育注入了新的活力,如DeepSeek和豆包。让孩子们可以通过简单的操作,训练自己的机器学习模型,实现图像识别、声音识别等功能,了解人工智能的基本原理和应用^[2]。

4 现实困境与突破路径

4.1 现存问题

教育公平问题是当前少儿编程教育面临的重要挑战之一。根据2023年教育部数据显示,农村编程课程覆盖

率不足12%，城乡之间存在着巨大的数字鸿沟。城乡儿童在编程教育机会上存在着明显的不平等。

师资瓶颈也是制约少儿编程教育发展的关键因素。编程教育需要既具备编程专业知识又掌握教育教学方法的复合型教师，但目前这类教师的缺口达76%。目前针对编程教师的培训体系还不够完善，也影响了教师专业素养的提升。

评价体系缺失也是当前少儿编程教育存在的问题之一。目前，大部分编程教育机构和学校的评价方式主要侧重于编程技能的考核，而忽视了对学生思维能力、创新能力和综合素养的评估。这种重技能轻思维的评估倾向，不利于全面了解学生的学习情况和发展潜力，也无法为教学改进提供有效的反馈^[3]。

4.2 解决方案

为了解决少儿编程教育存在的问题，需要建立分级课程标准。结合我国国情和学生的认知特点，制定适合不同年龄段学生的编程课程标准。分级课程标准应明确各阶段学生的学习目标、内容和要求，为教学提供指导，确保编程教育的系统性和科学性。

开发OMO混合式教学平台，将线上教学和线下教学有机结合。线上教学可以提供丰富的教学资源，如教学视频、在线编程环境、学习社区等，让学生可以随时随地进行学习。线下教学则可以通过面对面的授课、实践操作和小组讨论等方式，加强师生之间的互动和交流，提高教学效果。OMO混合式教学平台还可以利用大数据技术，对学生的学习行为和学习效果进行分析，为个性化教学提供支持。

构建“三维评价”体系，全面评估学生的学习情况。过程性评价通过记录学生的编程日志，了解学生在学习过程中的思维过程、问题解决策略和学习态度等，及时发现学生的学习困难和问题，给予针对性的指导。表现性评价通过学生的项目展示，评估学生的编程技能、创新能力、团队协作能力等综合素养，注重学生的实际表现和成果。发展性评价则通过建立学生的成长档案，跟踪学生的学习历程和发展轨迹，关注学生的个体差异和进步情况，为学生的长远发展提供支持^[4]。

5 未来发展趋势

随着家长对孩子早期教育的重视和编程教育理念的普及，少儿编程教育呈现出低龄化延伸的趋势。幼儿

编程教具不断创新，如Matatalab实物编程，通过让幼儿操作实物模块来编写程序，实现简单的指令控制，如前进、后退、转弯等。这种方式无需幼儿认识文字和数字，通过直观的实物操作，培养他们的逻辑思维和编程意识，为后续的编程学习打下基础。

未来，少儿编程教育将从“学编程”向“用编程学”的教育范式转型。编程将不再仅仅是一门独立的学科，而是作为一种工具和手段，融入到其他学科的教学之中。例如，在数学教学中，学生可以通过编写程序来模拟数学问题的解决过程，加深对数学概念和原理的理解；在科学教学中，学生可以利用编程来控制实验设备，采集和分析实验数据，培养科学探究能力。通过“用编程学”，学生能够更好地将编程知识与其他学科知识相结合，提高综合运用知识的能力，培养跨学科思维和创新能力。

结论

少儿编程教育不应仅仅被视为一种技术培训，而应上升为培养数字时代核心素养的基础工程。它对于儿童的认知发展、核心素养培育以及社会价值的延伸都具有不可忽视的重要意义。然而，当前少儿编程教育在发展过程中面临着教育公平、师资瓶颈和评价体系缺失等诸多问题，需要通过建立分级课程标准、开发OMO混合式教学平台和构建“三维评价”体系等措施加以解决。

展望未来，少儿编程教育将朝着低龄化延伸、技术融合以及教育范式转型等方向发展。为了实现少儿编程教育的可持续发展，需要构建政府主导、学校实施、企业支持、家庭参与的协同育人机制。政府应加大政策支持和资金投入，促进教育公平；学校应优化课程设置，加强师资队伍建设和实践平台。

参考文献

- [1]Wing J M. Computational Thinking[J]. Communications of the ACM, 2006.
- [2]教育部.义务教育信息科技课程标准(2022年版)[S].北京:北师大出版社.
- [3]Resnick M, et al. Scratch: Programming for All[J]. Communications of the ACM, 2009.
- [4]ISTE. Computational Thinking Competencies[Z]. 2021.