Detailed Table of Contents

Computer science education is undergoing a global transformation as digital technologies continue to redefine how knowledge is created, shared, and applied. Despite this evolution, many teaching practices in computing remain rooted in instructor-centered models that prioritize syntax and correctness over creativity, collaboration, and reflection. This chapter addresses this gap by presenting pedagogical innovations for the modern computer science classroom. Drawing on contemporary research in computing education and learning theories, it identifies ten interrelated themes and fifty practical strategies for effective instruction. Each theme is grounded in theory, contextualized through classroom examples, and supported by current literature. Collectively, these strategies advocate a shift from transmission-based teaching toward inquiry-driven learning that cultivates computational thinkers, creative problem solvers, and socially responsible innovators. The chapter concludes by discussing the implications of these pedagogical approaches for the future of computer science education.

This section highlights the importance of an interdisciplinary approach in teaching Computer Science (CS). This approach integrates various disciplines such as social sciences, mathematics, and technology to create holistic learning that is relevant to complex real-world challenges. An integrated curriculum model guides students in understanding the relationships between concepts from various fields, thereby increasing their engagement and deepening their understanding. This approach also encourages character and competency development, as well as supporting innovation in solving social and environmental problems. Although it offers many benefits, interdisciplinary implementation requires careful planning and sufficient resources, as well as addressing concerns related to technical depth. Overall, this approach is key to preparing graduates who are able to face global dynamics and advance the field of computer science in a sustainable manner.

This chapter explores the centrality of problem-solving in computer science education through the dual lenses of project-based learning (PjBL) and problem-based learning (PBL). While both approaches emphasize inquiry, collaboration, and real-world engagement, they differ in scope, structure, and outcomes. PjBL often immerses learners in extended projects that mirror professional practice, whereas PBL foregrounds open-ended problems that cultivate critical reasoning and adaptability. Anchored in the belief that computer science is not merely about mastering syntax or tools but about nurturing the ability to think computationally and creatively, the chapter argues for a “problem-solving first” philosophy in curriculum design. By examining practical applications, challenges, and emerging trends, it highlights how integrating PjBL and PBL can empower students to approach complexity with confidence, cultivate resilience in the face of ambiguity, and prepare for a rapidly evolving technological landscape.

This chapter proposes a pedagogy first architecture for computing assessment that couples formative routines with minimalist analytics and equity by design. It argues for alignment to disciplinary big ideas, visible success criteria, short revision cycles, and multimodal evidence including code histories, test rationales, diagrams, and oral defenses. Peer and self-judgment are cultivated as disciplined practices, supported by universal design and culturally sustaining tasks. Analytics remain small, interpretable, and privacy preserving, answering concrete instructional questions within hours. Capacity is built through coaching, lesson study, micro credentials, and calibrated observation, with change managed through protected schedules and toolkits. Comparative cases from K to 8, secondary, undergraduate, and reskilling contexts show feasibility and impact. An improvement science agenda links logic models to measurable gains in competence, fairness, and credible external signaling.

Computer science education is undergoing a profound transformation at the intersection of inclusivity, ethics, and rapidly evolving technologies. This chapter explores how innovative pedagogical approaches can effectively address the diverse needs of contemporary learners while simultaneously addressing the ethical, social, and societal challenges posed by technological advancements. Interdisciplinary models such as STEAM (Science, Technology, Engineering, Arts, and Mathematics), along with open-source tools and collaborative platforms, enhance experiential learning, creativity, and critical thinking skills. Central to this vision is the integration of ethical reasoning and responsible computing practices, equipping students to make principled, informed decisions in an increasingly automated and AI-driven world. The chapter concludes with actionable recommendations for educators, administrators, and policymakers, emphasizing the importance of professional development for computer science education.

Despite growing demand for computing professionals, women remain significantly underrepresented in computer science education and technology careers. This chapter presents a narrative review of literature on gender-inclusive pedagogies, examining classroom practices, curriculum design, assessment strategies, mentorship, and institutional initiatives that support female learners' engagement, persistence, and sense of belonging. Drawing on international and regional studies, the review identifies effective approaches such as collaborative learning, identity-affirming and real-world curricula, formative assessment, and structured support systems, while highlighting persistent gaps including Western-centric perspectives, limited intersectional analyses, and misalignment between policy rhetoric and classroom practice. Grounded in technofeminist theory, the chapter discusses implications for educators, institutions, policymakers, and future research aimed at advancing equity in computing education.

Computer science (CS) inclusive teaching provides an opportunity for everyone to learn in a way that is fair and accessible. Through the traditional approaches, neurodiverse and disadvantaged learners are often cut off due to the strictness of the instruction and the surrounding limitations on support. UDL (Universal Design for Learning) and CRP (Culturally Responsive Pedagogy) are the two major strategies for inclusive teaching that facilitate flexible, engaging, and relevant learning. UDL allows you to choose different ways to learn and to express your understanding, and CRP connects the content with the students' cultural backgrounds. Gallaudet University and AccessCSforAll are some of the initiatives that have been proving that learners are more engaged when using assistive tools and through the application of adaptive methods. Supported by educating CS for all, U.N. SDG 4, inclusivity

beyond surveillance-based models and adopt approaches that emphasize uniquely human cognitive capacities. These pedagogical strategies advance the field by shifting the discourse from reactive prevention to proactive, pedagogically aligned assessment design. In doing so, the chapter affirms that the future of programming assessment is not about resisting AI but about designing systems that assess human thinking over code output.

GenAI systems such as ChatGPT are increasingly discussed in programming education, but the ways in which the research literature conceptualizes and frames their role remain unclear. This chapter applies text mining to publications indexed in a leading academic database to map scholarly discourse on ChatGPT in programming education. Term frequency analysis, phrase pattern extraction, and topic modeling reveal four dominant themes: pedagogical implementation, student-centered learning and engagement, AI infrastructure and human–AI collaboration, and assessment, prompting, and model evaluation. The literature prioritizes classroom practice and learner interaction, with comparatively limited attention to assessment design and institutional governance. Across studies, ChatGPT is positioned both as a learning aid that supports explanation, feedback, and efficiency and as a pedagogical risk linked to overreliance, unreliable outputs, and academic integrity concerns. These findings support responsible integration and highlight the need for stronger assessment and governance mechanisms.

This chapter explores the use of intelligent learning systems to develop students' Python programming competencies in higher education. The research highlights how these systems provide automated assessment, personalized feedback, and gamified learning experiences that foster programming literacy, learner independence, and motivation. By demonstrating the effectiveness of intelligent systems in enhancing Python programming competencies, this chapter underscores the transformative role of artificial intelligence and machine learning in education. The results of this study clearly demonstrate that integrating intelligent learning systems into Python programming education significantly improves both the quality and efficiency of the learning process. The use of intelligent systems transforms traditional instructional models into adaptive, interactive, and learner-centered environments that align with the pedagogical demands of the digital age.

The advent of project-based learning (PBL) as a pedagogical method has made the Computer Graphics and Gaming course a practical, real-life learning experience to undergraduate computer engineers. Creating and producing their own video games, students combined the knowledge of graphics programming, storytelling, user interface design, animation, and game mechanics. It was a step-by-step process, starting with basic ideas and going into practice, and ending with demonstrations judged on technical implementation, creativity, interactivity and teamwork. The approach incorporated use of iterative design, peer feedback, and reflective practices and was inspired by the experiential learning cycle and multiple intelligences of Kolb and Gardner respectively. The chapter provides planning, implementation and evaluation approaches, discusses possible obstacles like resource shortage and

total of 582 documents been catalogued in Scopus for the period of 2014-2024 was used. The publications were analyzed using Biblioshiny for Bibliometrix and the results were used to identify gaps in the literature and guide future research directions on GAI's impact in education. The findings revealed the surge in publications and citations post 2020 can be attributed to large language models and AI coding assistants. Thematic and keyword analyses show that GAI has been applied primarily in programming education, intelligent tutoring, personalized learning, and assessment. These applications yield outcomes such as enhanced efficiency, student engagement, and scalability, yet also raise concerns about academic integrity, over-reliance, and algorithmic bias. GAI research is spearheaded by developed nations which poses fairness concerns, and simultaneously fosters advancement and moral quandaries in computer science pedagogy.