Journal of Contemporary Educational Research, 2025, Volume 9, Issue 7
http://ojs.bbwpublisher.com/index.php/JCER
ISSN Online: 2208-8474
ISSN Print: 2208-8466

Teaching Reform of the C Programming Language Course Integrating OBE Concept and AI Assistance under the New Engineering Background

Arzigul Ahat[1], **Gulnaz Alimjan**[1,2]*

[1]School of Network Security and Information Technology, Yili Normal University, Yili 844500, Xinjiang, China
[2]Yili Key Laboratory of Intelligent Computing Research and Application in the Yili River Valley, Yili Normal University, Yili 844500, Xinjiang, China

**Author to whom correspondence should be addressed.*

Abstract: With the continuous advancement of the New Engineering Education initiative, universities are raising the standards for cultivating engineering talents. C Programming Language, as a core course for computer science and related majors, plays a fundamental role in developing logical thinking, programming skills, and engineering practice. However, problems such as outdated content, weak practical connections, and single assessment methods still exist in current teaching, which affects both learning outcomes and students' skill development. Based on the outcome-based education (OBE) approach and supported by AI-assisted teaching tools, this paper proposes a reform plan focusing on teaching content, instructional methods, and evaluation systems. The goal is to enhance students' overall abilities and practical innovation skills, and to align the course more closely with modern industry needs.

Keywords: New Engineering; C Programming Language; Teaching reform; OBE concept; AI-assisted instruction

Online publication: July 31, 2025

1. Introduction

With the rapid development of information technology, engineering education is facing new challenges. In China, universities are actively promoting the construction of "New Engineering" programs to improve curriculum systems and enhance teaching quality [1].

C Programming Language is a fundamental course for computer-related majors. It plays a key role in helping students develop programming thinking and master basic algorithms. Due to its simple syntax, high efficiency, and close connection to hardware, C is widely used in embedded development, system software, and AI infrastructure, and also serves as a solid foundation for learning other programming languages.

However, many issues still exist in current teaching practices. The course content is outdated, examples lack relevance to modern contexts, and the teaching approach is mainly lecture-based. In addition, hands-on practice is limited, making it difficult for students to apply their knowledge flexibly. Traditional exam-focused assessment methods also fail to fully reflect students' true abilities.

At the same time, technologies such as AI and big data are gradually being introduced into classrooms. Tools like intelligent Q&A systems and automated grading are reshaping traditional teaching methods. More universities are also adopting the OBE concept, which places greater emphasis on improving students' real-world abilities.

This paper explores how AI technologies can be integrated into the teaching of C Programming Language under the New Engineering background. It proposes reform strategies in terms of teaching content, methods, and assessment, with the goal of enhancing students' programming skills and preparing them for future development needs.

2. Current teaching status and problems of C Programming Language

As a core foundational course for computer-related majors, C Programming Language plays an important role in developing students' programming skills and logical thinking. However, in practical teaching, several problems remain, including outdated content, limited teaching methods, and a weak connection between theory and practice.

(1) Outdated content that lacks relevance to engineering practice

Most current textbooks focus mainly on basic syntax and rarely cover topics commonly used in engineering, such as dynamic memory management, common data structures, and multi-threaded programming. As a result, students may understand syntax rules but struggle with complex tasks like embedded development or system design. The gap between textbooks and technological advances hinders students from developing systematic programming abilities.

(2) Traditional teaching methods with little classroom interaction

The "teacher speaks, students listen" model is still common. Classes lack interaction and hands-on practice. Teachers often focus on grammar explanations, while students passively absorb knowledge with low engagement. In real programming scenarios, students often lack a complete thinking process from problem analysis and function design to code implementation, making it difficult for them to handle complex tasks independently.

(3) Weak connection between theory and practice

Although the course includes lab sessions, they are usually concentrated in the later part of the semester and poorly integrated with theoretical lessons. Lab tasks are often simple and repetitive, and students tend to follow instructions without deep thinking or exploration. As a result, practical training becomes a formality, and students' hands-on skills do not improve significantly.

(4) Single assessment method with limited motivation

Current assessment still relies heavily on closed-book final exams, with regular grades mainly based on assignments and quizzes. Most questions focus on syntax and output prediction, lacking evaluation of programming and problem-solving abilities. This result-oriented evaluation cannot fully reflect students' true capabilities or inspire their continuous effort.

(5) Changing learning habits poses new challenges

Today's students are more used to fragmented and visual learning. However, C language has strict syntax rules and a complex debugging process, which often frustrates beginners when they encounter frequent errors. A fixed teaching pace also fails to address students' individual differences, leaving weaker students behind. How to combine smart teaching tools and personalized learning resources to improve adaptability and learning experience is now a key challenge in teaching reform.

3. Teaching reform strategies

3.1. Reforming teaching content

3.1.1. Layered instruction to match student learning pace

Following the "student-centered" approach, the course adopts a layered and progressive teaching model. Based on students' prior knowledge and cognitive patterns, the content is divided into three levels: basic, intermediate, and advanced. The basic stage focuses on core topics such as syntax, control structures, arrays, and functions. By combining theory with hands-on practice, this stage helps students master fundamental programming skills and develop good coding habits.

The intermediate stage builds upon the basics by introducing pointers and structures, and provides in-depth instruction on file operations and memory management. This phase aims to develop students' ability to handle more complex program structures and promote the advancement of their programming thinking.

The advanced stage focuses on key data structures such as linked lists, stacks, and queues, while integrating basic algorithms and concepts of concurrent programming. Students are encouraged to combine knowledge from all stages to carry out comprehensive practice projects, strengthening their problem-solving skills for complex programming tasks.

This layered structure sets clear goals for each stage, helping students avoid frustration caused by a sudden increase in difficulty. By gradually building knowledge and confidence, the course supports steady improvement in programming skills and reflects the principle of differentiated instruction.

3.1.2. Enhancing programming skills through real-world task-driven projects

To address the disconnect between theory and practice, the course introduces task-driven projects based on real-world application scenarios. By completing specific functional tasks, students acquire programming skills in a more meaningful and practical way. Compared to traditional grammar-based instruction, task-driven learning better stimulates student interest and aligns with the goals of engineering-oriented training [2].

For example, in the early stage, students work on a "Student Grade Management System" to practice array processing, function calls, and file operations. In the intermediate stage, project difficulty increases, such as a "Library Management System" involving structure definitions and data state control, and a "Bank Account Management Program" focusing on pointers and dynamic memory allocation, which require stronger abstract thinking and logical reasoning.

These tasks cover the core content of the course. Through project-based learning, students not only improve their programming abilities but also strengthen their problem-solving skills, laying a solid foundation for future engineering work or academic research [3].

3.1.3. Updating teaching resources to reflect technological advances

As C language continues to expand its applications in areas such as the Internet of Things, artificial intelligence, and embedded systems, teaching content must also evolve accordingly. If the course still relies on programming styles from decades ago, it will no longer keep pace with industry trends. Therefore, instructors should update course materials flexibly based on technological developments [4]. AI tools like DeepSeek or Doubao can be used to explore innovative and practical examples, from which suitable cases can be selected and introduced into the classroom.

For instance, presenting a basic case of “implementing a neural network structure in C” or designing a small program for sensor data collection can be more engaging than teaching grammar rules alone. Such examples not only demonstrate the real-world value of C programming but also help students gain early exposure to its practical use in various systems.

3.2. Reforming teaching methods

3.2.1. Introducing project-based learning (PjBL)

To shift students from passive listening to active doing, this course integrates PjBL throughout the teaching process. Unlike traditional chapter-by-chapter lectures, instructors design a series of progressively challenging mini-projects based on the course schedule and students’ learning progress, allowing them to learn by doing [5].

At the beginning, tasks are relatively simple (for example, writing a basic student information management program) to help students practice input/output, conditional statements, and control structures. As the course advances, project complexity increases with the addition of file operations, modular functions, and basic system design. In the final stage, students are assigned more challenging projects, such as building a linked list management module, which guides them toward real-world development practices and coding standards.

This hands-on learning model not only helps students better understand theoretical concepts, but also cultivates their programming mindset through continuous practice. Compared to passive learning, students become more motivated to identify and solve problems independently, which prepares them for future work environments.

3.2.2. Integrating flipped classroom and blended learning

To enhance classroom interaction and student engagement, the course adopts a combination of flipped classroom and blended learning approaches. Before class, instructors upload learning materials and self-assessment tasks to an online platform, allowing students to study foundational content at their own pace. This frees up in-class time for Q&A sessions, case analysis, and hands-on practice.

The teaching is problem-driven, incorporating explanations of common mistakes and small project-based exercises to help students deepen their understanding through real-world application. This shift in teaching approach makes the classroom more open and interactive, encouraging active participation. Meanwhile, the platform can track learning duration and task completion, enabling instructors to adjust the pace or offer differentiated guidance based on students’ learning status.

3.2.3. Using AI platforms to improve learning feedback efficiency

In programming courses, students often feel frustrated when they fail to debug code or repeatedly encounter errors, which can lower their motivation. To address this, the course introduces an AI-assisted platform that automatically analyzes students’ submitted code. When syntax errors, logic issues, or formatting problems are detected, the system highlights them in real time and provides suggestions for improvement. This helps students quickly locate mistakes, reduces time spent on repeated debugging, and improves their learning experience.

In addition to error correction, the platform records students’ practice data and generates learning reports that include common error types, submission frequency, and more. Instructors can use this data to monitor student progress and adjust teaching strategies accordingly. It also enables more targeted guidance based on students' individual learning levels.

3.3. Teaching evaluation reform

3.3.1. Improving evaluation structure to reflect the entire learning process

Traditional evaluation methods often rely too heavily on final exam results, which fail to fully reflect students' actual learning performance. To address this issue, the course adopts a more comprehensive assessment system that covers the entire learning process from multiple dimensions.

The evaluation system consists of four components designed to cover the entire learning process: daily assignments and class participation (15%) focus on students' engagement and learning habits; project outcomes (30%) assess practical skills and problem-solving ability; learning process monitoring (25%) uses platform data such as study time, practice frequency, and error correction to track student progress; and the final exam (30%) combines open-book and open-ended questions to evaluate comprehensive understanding and knowledge application.

This assessment system goes beyond a single-score approach. By emphasizing learning process and practical abilities, it encourages students to shift from exam-oriented thinking to application-based learning and better demonstrates their progress and skill development throughout the course.

3.3.2. Introducing peer review to strengthen teamwork awareness

In team-based projects, relying solely on instructor evaluation often fails to reflect each student's actual level of participation. To enhance fairness and guidance, the course incorporates peer assessment and self-evaluation mechanisms.

After completing a project, students are required to anonymously evaluate their group members based on task execution and collaboration attitude, while also submitting a personal reflection to summarize their role and learning experience. When grading, instructors consider project outcomes, peer review results, and self-assessments to more objectively evaluate each student's contribution and engagement. This approach encourages students to develop self-awareness, active listening, and a sense of responsibility in teamwork—laying a foundation for future collaborative projects.

4. Conclusion and outlook

The recent adjustments to the C Programming Language course focused on content structure, teaching methods, and assessment strategies. To reduce reliance on traditional lectures, more hands-on activities were introduced to encourage students to practice and think independently. Evaluation is no longer based solely on exam scores; instead, learning progress is monitored to provide a more comprehensive understanding of student performance.

The results so far have been positive: student engagement has increased, and task completion has improved. The use of AI tools has also made it easier for instructors to track learning status, identify problems early, and make timely adjustments. Although the reform is still in the exploratory stage, initial feedback has been encouraging.

Moving forward, the course will continue to follow technological developments, introduce more practical projects, and promote collaboration and critical thinking. Teaching reform is an ongoing process—it requires continuous experimentation and reflection to make the course more relevant to real-world needs and better support student growth.

Funding

This study was funded by Xinjiang Natural Science Foundation of China (2023D01C52); University Key Project (2023YSZD004).

Disclosure statement

The authors declare no conflict of interest.

References

[1] Li G, Zhao F, Zhou Y, 2025, The Teaching Reform and Practice of Professional Courses for Engineering Students. Journal of Contemporary Educational Research, 9(5): 290–295.

[2] Gan Y, 2024, The Organic Integration of Innovation and Entrepreneurship Education with Ideological and Political Education—A Case Study of the C Programming Course. International Journal of Educational Teaching and Research, 1(1).

[3] Amisha S, Nancy A, Anil T, 2024, Modern Approach to C Programming: Exploring the Foundations of Problem-Solving through C Programming (English Edition), BPB Publishers, India.

[4] Zu B, Wang H, Chen H, et al., 2025, Adapting High-Level Language Programming (C Language) Education in the Era of Large Language Models. Journal of Contemporary Educational Research, 9(5): 264–269.

[5] He Y, He Y, 2024, Research on C Language Programming Case-Assisted Teaching Based on BUGs Exclusion. Curriculum and Teaching Methodology, 7(5).