2024 Volume 2, Issue 2
Computer Simulation in Application
ISSN: 2630-4597 (Online)

A Spiral Learning Model for Computer Practical Education Based on Multidimensional Practice

Xiaolin Zhao*, Jingfeng Xue, Yong Wang, Fengnian Zhao, Tianyu Huang, Shuoying Chen
School of Computer Science, Beijing Institute of Technology, Beijing 100081, China

***Corresponding author:** Xiaolin Zhao, zhaoxl@bit.edu.cn

Abstract:

In view of the problem that the cultivation of practical skills in computer education in universities is mainly focused on course assignments or ordinary practical courses, and has not formed a partial order skill progression like the theoretical course group, this paper proposes a spiral learning model for computer education based on multidimensional practice. This model takes the "craftsmanship spirit" as the core of the ideological and political construction of the integration of industry and education, and "ideological and political guidance, problem orientation" as the characteristics of the course group. It expounds on how to construct a multidimensional practical spiral model through the internal circulation of the course with the learning method of "explaining, doing, demonstrating, and improving," and the ascending external circulation among different grades with the learning objectives of "mastering, understanding, researching, and innovating." Finally, the teaching effect is illustrated through the evaluation of teaching data in the past three years, and suggestions for continuous improvement are provided.

Keywords:

Computer practical teaching
Multidimensional practice
Spiral learning
Explaining, doing, demonstrating, and improving
Mastering, understanding, researching, and innovating

Online publication: December 27, 2024

1. Introduction

Currently, computer networks have penetrated deeply into people's lives. Smartphones have become internet terminals, smart homes and robots are realizing scenes from science fiction movies, and health codes are assisting in the fight against pandemics. All of these are inseparable from the labor of "code farmers." However, the work of "code farmers" is not limited to the coding process. It involves more software design, testing, and higher-level research, development, and innovation. This

requires undergraduate computer education in universities to focus not only on coding practical abilities but also on enhancing feedback and improvement from practice to theory. The training mode of computer talents has always been the focus of teaching and research for university teachers [1]. In recent years, innovative talent training has become the main direction of teaching research [2]. However, in current university computer education, most practical ability improvement channels are through course practices [3] or separate practical courses [4]. These focus on improving horizontal abilities within the course but lack the joint application of vertical course knowledge and progressive practical goals. There are still deficiencies in students' progressive computer ability improvement, which can cause bottlenecks in their practical ability improvement and prevent them from receiving "innovation incentives." This, in turn, can affect the improvement of abilities within the course and make it difficult to achieve deeper ideological and political goals [5]. To address this issue, as a double first-class university, we attempt to integrate three years of vertical practical courses to form a multidimensional practical spiral model of computer education, consisting of an intra-course cycle with an "explaining, doing, demonstrating, and improving" learning approach and an extra-course ladder with "mastering, understanding, researching, and innovating" learning objectives.

2. Spiral learning model of computer education based on multidimensional practice

The spiral learning model based on multidimensional practice is shown in **Figure 1**. This model is not targeted at a single course, nor is it limited to practical courses. It is a condensation and refinement of four years of undergraduate practical education.

This model directly involves four practical courses across three academic years, including the second-year courses "Basic Training in Internet Application Development" and "Basic Training in Software Engineering," each worth 1.5 credits; the third-year course "Comprehensive Practice in Software Engineering," worth 3 credits; and the fourth-year course "Professional Training in Software Engineering," also worth 3 credits. Each course spans three weeks, totaling 9 credits. The theoretical foundation of these four practical courses is derived from a year of theoretical courses taken by the students, covering the main knowledge system of computer education. The main thread of the spiral model is "ideological and political guidance, problem orientation," which constructs the framework of the spiral learning model. "Explaining, doing, demonstrating, and improving" constructs the spiral learning process, while "mastering, understanding, researching, and innovating" builds a progressive evolution of abilities layer by layer. The combination of these three enriches the connotation of the spiral learning model.

(1) We use ideological and political guidance to drive learning motivation and build a new channel for ideological and political construction, guiding students to practice the "craftsmanship spirit," "contract spirit," and "initiative spirit." We orient learning progress around problems, following the principle that "practice is the only criterion for testing truth." Through analyzing and solving problems, we conduct scientific research exploration, implementing ideological and political education in the classroom [6]. Software design and programming implementation are also forms of labor, and the craftsmanship spirit is primarily a labor spirit. We establish a sense of mission to serve the country through software, and through careful guidance and strict requirements, we complete project tasks with high quality, practicing the

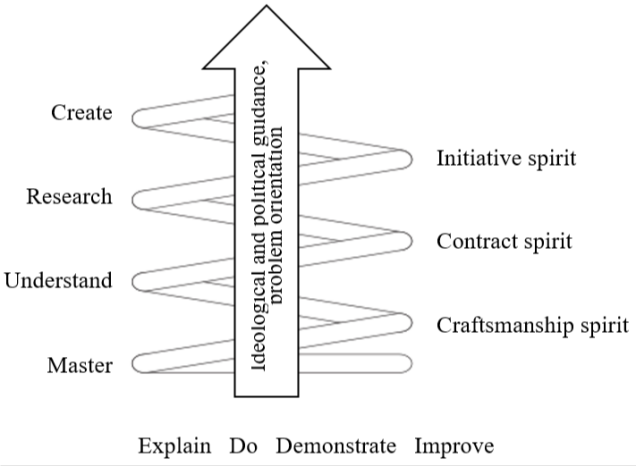

Figure 1. Spiral learning model

dedication, meticulousness, and excellence of the craftsmanship spirit.

(2) Each layer of the spiral learning model constructs the learning process according to "explaining, doing, demonstrating, and improving," and different layers progressively advance according to "mastering, understanding, researching, and innovating," forming a spiral, three-dimensional, and progressively ascending learning model. In this model, each practical course is both a comprehensive practical test of the theoretical learning from the previous year and an incentive to guide students' learning in the following year. By longitudinally connecting multiple practical courses across three academic years, we have constructed a spiral learning model and practical curriculum system suitable for the undergraduate stage of research-oriented universities, enabling multiple courses to form a progressive relationship in content and ability.

(3) Through the implementation of the spiral learning model and practical curriculum system, we adopt a horizontal cyclic teaching process control of "explaining, doing, demonstrating, and improving" and a vertical progression of abilities through "mastering, understanding, researching, and innovating." Students conduct research in practice and innovate from research, cultivating the "initiative spirit."

It should be emphasized that the horizontal and vertical aspects of this spiral model are inseparable and integral parts. They are the flesh and blood of the spiral model, and every aspect requires careful design and implementation.

3. Building an ideological and political construction channel for the integration of industry and education with "craftsmanship spirit" as the core

Scientific research practice is a practical activity guided by theory, and it necessarily requires a spirit throughout the process. Combining the "practical labor" characteristics of practical courses, we carefully design the evaluation of labor for second, third, and fourth-year students [3], and propose an ideological and political construction channel with "craftsmanship spirit" as the core, forming a distinctive feature of "ideological and political guidance, problem orientation."

The "craftsmanship spirit" is first and foremost a labor spirit. Relying on the educational concepts of OBE [7] and CDIO [8], as well as the requirements of engineering education certification [9], we view computer practical activities as a form of labor. Engels pointed out that "real labor...begins with the manufacture of tools." The manufacture of tools is the problem addressed in computer practical courses. The second-year practical courses focus on using existing tools for labor, such as building websites; the third-year practical courses imitate the manufacture of tools, such as developing compilers; and the fourth-year practical courses involve creating new tools, such as analyzing network protocols and traffic to measure network security status. Through the design of course learning objectives and labor processes, we help students understand the characteristics of engineering education [10], explore professional development directions, reflect on professional missions, establish lofty aspirations to serve the country through software, stimulate students' enthusiasm for learning and labor, and cultivate top talents [11].

The "craftsmanship spirit" embodies the dedication and initiative spirit of professional labor. In the second-year practical courses, we guide students to understand the current gaps and urgent needs of China's software industry, advancing their cognition of software understanding. The third-year practical courses raise questions targeting the weak links of China's basic software, guiding students to join the pursuit, such as designing and implementing small databases and comparing them with domestic and foreign commercial databases. The fourth-year practical courses focus on innovation, transforming the university's scientific research platform into a teaching innovation platform [12], combining innovation with dedication to form the driving force for students' efforts.

The "craftsmanship spirit" represents a meticulous and excellence-seeking contract spirit. Emphasizing details and pursuing perfection are key elements of the "craftsmanship spirit" and manifestations of engineering capabilities [13]. Through the signing of classroom written contracts, quantifiable and evaluable agreements are formed in terms of labor objectives (project goals),

time requirements, funding, software functionality, and performance. In the evaluation of practical courses, the perfection of software is assessed through details such as software human-machine interfaces, interfaces, boundary value testing, functional stability, and computational efficiency. Team collaboration and engineering management are evaluated based on research and development models, process management, document preparation, teacher-student interaction, budgeting and final accounting, software demonstration, and promotion [14]. This makes the "craftsmanship spirit" a spiritual pillar for software engineering to transition from a "craft" to an "industry."

"Ideological and political guidance, problem orientation" helps students open the door to scientific research, stimulates creative thinking, and cultivates innovative talents. Scientific theoretical learning and engineering practice are reflected through a spiral practical process. Practical topics are selected based on tasks directly related to national and social needs, introducing research tasks that align with university characteristics. A problem-oriented teaching model is implemented [15], establishing a new ideological and political model that cultivates students' sense of mission toward professional cognition and their sense of responsibility towards acquiring independent intellectual property rights for significant basic software.

4. Constructing the "explaining, doing, demonstrating, and improving" intra-course learning process

"Explaining, doing, demonstrating, and improving" is a complete intra-course learning cycle implemented in each course. While the objectives of "explaining, doing, demonstrating, and improving" may vary across courses, the adopted approach is similar. A complete cycle of "explaining, doing, demonstrating, and improving" includes four processes: course design, strategy, teaching practice, and evaluation, as shown in **Figure 2** [16].

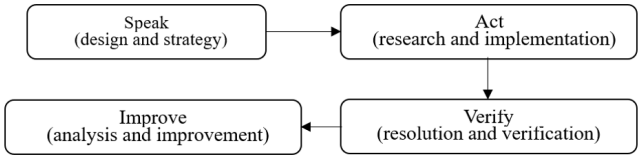

Figure 2. The four processes of "explain, do, demonstrate, improve"

Taking the senior year software engineering professional training course as an example, the work of these four stages is as follows:

(1) Be able to complete the practical process of software engineering projects, write various software engineering documents, and evaluate them. During the practical process, conduct scientific research on complex software engineering problems, demonstrating basic scientific research quality and ability (explain).

(2) To achieve the above goals, students should complete and submit tasks including requirements analysis, system design (incorporating software engineering methods), software development (incorporating new technology learning), the entire process of system testing, and promotion (English presentation). Based on the topic, students must conduct a series of scientific research processes such as researching materials, discussion and analysis, system modeling, and problem-solving, and submit a research report. The guidance process focuses on cultivating students' exploratory spirit, scientific thinking, practical ability, and innovative ability. Practical teaching should not only impart experimental skills and operational abilities, but also be positioned as systematic imparting and learning training of experimental science and skills, as well as practical and innovative abilities (do).

(3) Through the collection and analysis of student submissions, it is confirmed that students have completed the general process of complex software engineering development, can write software engineering documents, and can perform practical work such as requirements analysis and pattern design. Through research and analysis, students propose their own solutions to the problems existing in the research process (self-study, referring to the latest relevant theoretical research results), analyze individual student situations and overall scores, and make longitudinal comparisons with previous teaching situations (demonstrate).

(4) By analyzing student homework and research

reports, longitudinally comparing the teaching process, scores, teaching deficiencies, and countermeasures of the previous session, the current teaching process is timely revised, and deficiencies in the current teaching process are analyzed. Improvement strategies for the next time are provided in three aspects: theoretical mastery, complex software development, and related theoretical research (improve).

The above four processes constitute a cycle, continuously improving the quality of the course in a spiral manner. By designing the complete process of "explain, do, demonstrate, improve" for each expected learning outcome, and continuously adjusting it based on actual situations during implementation.

To encourage students to conduct research and innovate, they are allowed to apply for innovation scores based on the originality and novelty of their research results, which will be counted as part of the total course score.

In the design of the "explain, do, demonstrate, improve" feedback standards, apart from emphasizing the three major feedback questions ("Where am I going?," "How do I get there?," and "Where to next?"), the following principles are also followed: (1) Feedback focuses on tasks rather than students; (2) Provide well-designed feedback that describes "what," "how," and "why"; (3) Feedback information is specific and clear, directly changing the performance evaluation criteria; (4) Feedback should be as simple and objective as possible, integrating feedback from both instructors and students; (5) Emphasize immediate feedback for low-achieving students and delayed, innovative feedback for high-achieving students.

5. Constructing a spiral learning model of "master, understand, research, innovate" across courses

"Master, understand, research, innovate" is a requirement for students' abilities designed according to their year of study (as shown in **Figure 3**). Each course mainline adopts case-based teaching, with specific methods varying among courses. The sophomore-level courses on basic training in internet application development and software engineering require students to transition from "master" to "understand" abilities. The junior-level course on comprehensive software engineering practice requires students to transition from "understand" to "research" abilities. The senior-level course on software engineering professional training requires students to make the transition from "research" to "innovate" abilities.

5.1. "Master" to "understand"

The meaning of "master" to "understand" is the transition from being able to program to understanding software, with the targeted career being an engineer. Students have completed learning C language in their first year of college and can "master" programming, but they lack an overall concept of software. The meaning of "understand" is to grasp the basic structure of software, understand basic software design methods, and comprehend the basic uses of software.

Basic Training in Internet Application Development guides students to design and develop an Internet application, covering knowledge areas such as the basic structure of the Internet, website structure, CSS, JavaScript, HTML5, and Android. This allows students

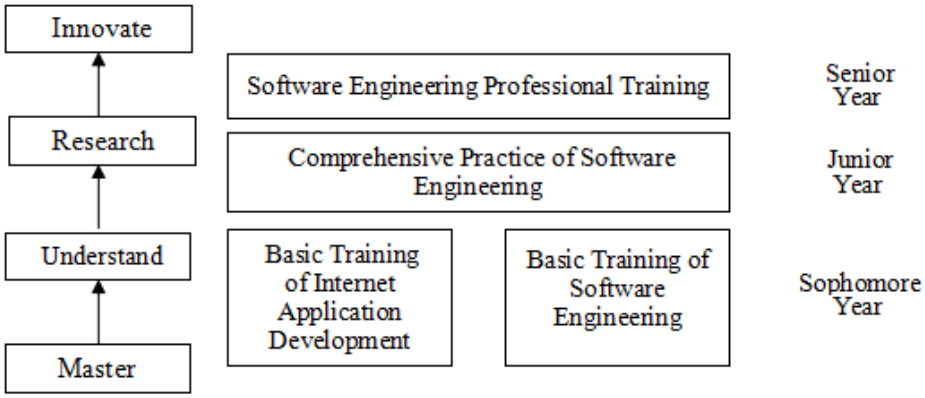

Figure 3. "Master, understand, research, innovate" ascension diagram

to gain a preliminary understanding of the Internet, software architecture, and the software design process from a macro perspective. Basic Training in Software Engineering introduces students to new technical topics, including data, intelligence, networking, and graphic visualization, and guides them to complete comprehensive project selections based on their personal interests. This helps students to gain an initial understanding of software engineering and comprehend the software engineering process. By adopting case-based teaching and a multi-task-driven approach, students are encouraged to transition from autonomous needs (the "master" software stage) to competency needs (the "understand" software stage).

To address the issues faced by these two courses, such as the simplicity of the content, limited class hours, and the large number of students, which make it difficult to stimulate interest in learning, provide sufficient class hours, and offer personalized teaching, we have rebuilt the teaching content. The teaching content consists of three parts: strengthening programming, practical areas, and understanding software. The courses introduce advanced technologies and multi-domain software technologies, including big data, artificial intelligence, virtual reality, etc., guiding students to shift their focus from programming-related case studies to practical application-oriented software engineering. The result is that students "understand" the basic process of software development and expand their knowledge system.

5.2. "Understand" to "research"

The meaning of "understand" to "research" is the transition from understanding software to researching software, conducting research in practice, with the targeted career being a scientific researcher. After completing the "understand" training, students begin to learn professional courses in their second year of college and start to understand the basic principles of computers and software from a theoretical perspective. However, there is a huge gap between theory and implementation, which requires research.

The Integrated Practice Course in Software Engineering, offered in the third year, allows students to learn new knowledge and research specific problems through means such as literature study, while integrating professional course knowledge. The topics mainly address the shortcomings of China's three major basic software areas. The related knowledge involved in the course projects, such as "Compilation Principles," "Operating System Principles," and "Database Principles," are all content that will be learned in subsequent courses.

For example, a topic like a C-language-like interpreter requires an understanding of C language's operational rules and basic programming principles (the "understand" software stage). However, knowledge such as the QT development environment and debug principles are not covered in regular college courses. These knowledge points require students to quickly learn independently based on project requirements. To complete the interpreter, students need lexical analysis and syntax analysis from "Compilation Principles," which involves theoretical exploration (the "research" stage). This enables students to comprehensively apply the basic theories of software engineering and computer knowledge, organically integrating theoretical guidance with development practice, combining engineering goals with independent innovation, and effectively coordinating personal strengths with team collaboration. Through internships, students cultivate a pragmatic mindset, a down-to-earth work style, and meticulous engineering abilities, achieving the goal of combining theory with practice and integrating engineering with innovation.

5.3. "Research" to "innovate"

The meaning of "research" to "innovate" is to innovate from research, targeting the career of a scientist. The Software Engineering Professional Training course has two threads. One is to formally complete projects in practice according to software engineering methods, and the other is to closely follow national strategic needs and innovate in research in terms of content. For example, combining the cutting-edge research directions of cyberspace security and big data, a topic such as "Windows Malicious Code Classification Based on Machine Learning" is given. This requires selecting appropriate features and classification algorithms based on the static analysis results of given samples (research stage), implementing the classification of given codes, and making necessary improvements to the algorithm

based on specific datasets (innovation stage).

For the first thread, the course explains the background knowledge of application fields, completes the requirements analysis of complex software systems, completes the software engineering project practice process in terms of results, writes various software engineering documents, and evaluates them. In the process of practice, scientific research is conducted on complex software engineering problems, completing the general scientific research process and exercising students' basic scientific research quality and ability. The course emphasizes process management, involving team setting, project management, and economic decision-making. It encourages innovation, involving technological innovation, scientific research, and engineering innovation, and sets up a dedicated innovation score. A software engineering process plan needs to be developed, and the training and research process should have a clear plan that can be accurate to the day and retain redundancy. Appropriate software process tools are used to support training and research work, meeting project management requirements.

For the second thread, the topics for practical training come from teachers' vertical scientific research tasks, closely following major national needs (such as the National Key Research and Development Program and the National Natural Science Foundation of China projects) and the latest cutting-edge technologies (such as network security, artificial intelligence, big data, and digital media). Fragments with more practical links are extracted to form relatively complete research topics. Firstly, the topics should meet the basic requirements of the course content and the requirements of complex software engineering problems. Secondly, they should have a certain level of difficulty to stimulate students' enthusiasm, and it should be confirmed that they can be basically completed during the short semester. The research can be theoretical or technical, but it must be carried out with practice as the carrier and practical training as the result. Finally, through the combination of form and content, an innovative environment for students is provided, and students are encouraged to think with a "pioneering" spirit in practice through the encouragement of innovation scores.

In this way, there are innovation scores in the form of research practice, and innovation determines whether a student's performance is excellent or good. In terms of the content of research practice, closely following major national needs (such as the National Key Research and Development Program and the National Natural Science Foundation of China projects) and the latest cutting-edge technologies (such as network security, artificial intelligence, big data, and digital media) ensures the practical value and originality of innovation.

6. Evaluation of teaching effectiveness and continuous improvement of the spiral model

6.1. Evaluation of teaching effectiveness

The courses involved in this achievement have been running for four years according to this spiral learning model, with nearly 1,000 students participating. Among them, over 300 students from two grades have experienced a complete spiral learning model, while the remaining students have completed one or two cycles.

Based on the analysis of graduation project topics in the past three years, the proportion of students choosing theoretical research has increased from 24% to 35%. The analysis of dissertations shows that the percentage of students demonstrating innovation has risen from 24% to 40%, indicating the positive impact of this achievement on research and innovation. In terms of plagiarism checks, the proportion of dissertations with a repetition rate exceeding 20% has dropped rapidly from 20% to 8%, suggesting that the "craftsmanship spirit" has subtly influenced students.

Over the past three years, the curriculum system's OBE support scores (with a maximum score of 1) have been above 0.8 for all items, with scores related to research practice activities averaging around 0.85. This indicates stable cultivation of students' practical abilities without being affected by scientific research training and innovation. The high research scores reflect improved scientific research capabilities among students, achieving the curriculum system's goal of integrating research into practice and fostering innovation through research. In terms of practical results, each teaching activity generates over 1GB of research practice documents and codes, forming a valuable accumulation for the learning model.

After completing the comprehensive spiral learning model training, students' overall abilities have significantly improved. Some students have even discovered their research interests and begun participating in scientific research work in teachers' laboratories. Often, they can determine their research direction for their master's degree after completing the practical courses in their fourth year. Those who are recommended for postgraduate studies actively contact their mentors to start their graduation project tasks and postgraduate research topics ahead of time.

6.2. Continuous improvement

The "explain, do, demonstrate, improve" approach represents a wave-like spiral progression, while the "master, understand, research, innovate" methodology reflects a spiral ascension through negation. Combining these two approaches constitutes the spiral learning model. This model is formally a result of the continuously increasing demands of practical courses at research universities and is essentially an inevitable outcome of philosophical guidance on natural science methodologies.

The model, which currently only tracks until graduation design, cannot yet fully illustrate the extent of improvement in students' innovation abilities. This will be analyzed and continuously improved in subsequent educational reform practices. The model also faces the requirement of its own continuous improvement, necessitating both horizontal learning process enhancements and vertical connectivity improvements. Starting from observing teaching details, careful adjustments to course schedules and case studies, as well as the introduction of new tools and methods, are essential. This approach aims to drive technological, teaching, theoretical, and scientific advancements through a two-way feedback spiral of continuous improvement.

7. Conclusion

Ideological and political guidance is the essential connotation of the practical curriculum system, while the "craftsmanship spirit" serves as its guiding principle. Problem orientation lays the foundation for the curriculum system, scientific research training constitutes its methodology, and innovative practice represents its ultimate goal. These elements—ideological and political guidance, problem orientation, practice, scientific research, and innovation—form the core components of the curriculum system and establish the framework for an innovative practical curriculum system.

Spanning three academic years, the curriculum system aids in guiding students to deepen their theoretical course learning and consciously expand their knowledge structure. It also lays a solid foundation of practical abilities for students' graduation projects and postgraduate innovation. The model's innovativeness lies in its ability to guide students toward self-improvement through a spiral ascension model, embodying the "spirit of labor." It emphasizes that "teaching someone to fish is better than giving them a fish," forming an operable practical system with innovation ability cultivation as the overarching goal of practical results. This approach aims to guide students from the "realm of necessity" to the "realm of freedom" in their practical thinking, potentially serving as a valuable reference for practical education in other disciplines.

Disclosure statement

The authors declare no conflict of interest.

References

[1] Shi J, Chen J, Xue J, et al., 2019, The Top-Notch Innovative Talent Training Mode of the Five-in-One Software Industry. Computer Education, 2019(5): 110–114.

[2] Yu L, Wang H, Liu S, 2020, Research on the Cultivation of Excellent Top Talents in Computer Science. Software Guide, 19(2): 157–159.

[3] Du S, Yang C, Liu Y, 2022, Exploration of Formative Evaluation Methods for Network Programming Technology Courses.

Computer Education, 2022(1): 148–151.

[4] Long C, 2010, Course Group Construction: The Path Choice for Teaching Reform of College Courses. Modern Education Science: Higher Education Research, 2010(2): 139–141.

[5] The Central People's Government of the People's Republic of China, 2017, The Central Committee of the Communist Party of China and the State Council issued the "Opinions on Strengthening and Improving Ideological and Political Work in Colleges and Universities Under the New Situation," viewed July 4, 2023, http://www.gov.cn/xinwen/2017-02/27/content_5182502.htm

[6] Zhang R, Mestre P, Gao W, et al., 2022, Exploring the Infiltration Mode of Ideological and Political Education in Science and Engineering Practice Courses. Computer Education, 2022(6): 5–9.

[7] Yu C, Jiang Y, Chen L, et al., 2022, Discussion and Practice of OBE Teaching Reform in Software Engineering Courses. Computer Era, 2022(6): 104–107, 111.

[8] Hu Z, Ren S, Wu B, 2010, Constructing an Integrated Curriculum Teaching Model Based on the CDIO Concept. China Higher Education, 2010(22): 44–45.

[9] Lin J, 2015, Engineering Education Certification and Engineering Education Reform and Development. Research in Higher Engineering Education, 2015(2): 10–19.

[10] Liu L, He L, 2021, Research and Practice of Blended Teaching Mode for Engineering Experiments Based on the Concept of Engineering Education Professional Certification. China Modern Educational Equipment, 2021(8): 90–92, 98.

[11] Wu J, Xia X, 2020, Research on the Construction Plan of Top Talent Training Bases in Universities under the Background of "Double First-Class": Taking Computer Top Talent Training as an Example. Computer Knowledge and Technology, 2020(35): 87–88.

[12] Zhang Z, 2012, Creating a Practical Teaching Platform to Enhance Students' Innovation Ability. China Higher Education, 2012(6): 25–27.

[13] Wang A, Du X, 2022, Practical Teaching Reform with Organic Integration of Innovation, Entrepreneurship, and Professional Practice. Computer Education, 2022(3): 130–133, 138.

[14] Shen Z, Guo Y, 2017, Exploring the Assessment Methods of Computer Programming Practice Courses. Education Observation, 2017(13): 101–102.

[15] Li T, 2021, Research on the Design of Problem-Oriented Teaching Mode. Education Modernization, 8(34): 170–174.

[16] Gao X, 2022, Deep Thought, Aggregation, and Discrimination: Deep Classroom Observation and Teaching Practice. Computer Education, 2022(9): 3–4, 6.