The OSCAR PUBLISHING Services
American Journal of Applied Science and Technology

VOLUME Vol.05 Issue 12 2025
PAGE NO. 205-210
DOI 10.37547/ajast/Volume05Issue12-37

Developing Algorithmic Thinking Skills Of Specialized School Students: A Case Study Of Mingbulak District

Meliyeva Iroda Xamidovna
Computer Science and Information Technology Teacher At specialized school in Mingbulak, Uzbekistan

Received: 27 October 2025; **Accepted:** 18 November 2025; **Published:** 24 December 2025

Abstract: This article investigates the effectiveness of practice-oriented instruction in developing algorithmic thinking and digital competencies among students of a specialized school in Mingbulak district. The study was conducted with 8th–9th grade students divided into experimental and control groups. A six-week instructional intervention based on practical programming tasks, mini-projects, and debugging activities was implemented in the experimental group. Pre-test and post-test results demonstrate a significant improvement in students' problem-solving abilities, algorithmic accuracy, and coding logic in comparison with traditional instruction. The findings confirm that systematic integration of practical tasks enhances students' computational thinking and learning motivation in informatics education.

Keywords: Informatics education, algorithmic thinking, computational thinking, digital competence, practical tasks, specialized school, mini-projects.

INTRODUCTION:

The accelerating pace of digitalization has fundamentally transformed contemporary education, placing informatics at the center of school curricula worldwide. Beyond its technical dimension, informatics serves as a powerful medium for developing learners' analytical thinking, logical reasoning, and problem-solving abilities. In modern educational paradigms, digital competence is increasingly viewed as a core life skill, enabling students to function effectively in technology-rich academic, professional, and social environments. Consequently, informatics education is no longer limited to the transmission of theoretical concepts but is expected to cultivate students' ability to apply algorithms and computational principles to authentic, real-world problems.

Current trends in education emphasize learner-centered and competence-based approaches, where knowledge acquisition is closely integrated with practical application. Within this framework, algorithmic thinking is regarded as a foundational component of computational thinking, encompassing skills such as problem decomposition, pattern recognition, abstraction, and step-by-step solution design. However, despite the growing recognition of these skills, traditional informatics instruction often remains predominantly theory-oriented, with limited opportunities for students to engage in meaningful hands-on activities. As a result, learners may demonstrate satisfactory performance in written tests while struggling to design functional algorithms or implement solutions independently.

Specialized schools, particularly those with a focus on science and technology, occupy a strategic position in addressing this challenge. These institutions are designed to identify, support, and further develop students with advanced cognitive abilities and strong interest in technical disciplines. Nevertheless, classroom observations conducted in a specialized school in Mingbuloq district indicate that even academically capable students frequently encounter difficulties when transitioning from theoretical understanding to practical implementation. Common challenges include constructing logically consistent

algorithms, structuring code effectively, debugging errors during program execution, and adapting learned concepts to novel tasks. Such difficulties highlight a discrepancy between conceptual knowledge and practical competence.

This gap underscores the necessity of rethinking instructional strategies in informatics education, particularly in specialized learning environments. A growing body of pedagogical research suggests that practice-oriented instruction—characterized by hands-on programming tasks, mini-projects, collaborative problem-solving, and reflective activities—can significantly enhance students' algorithmic and computational thinking. Mini-projects, in particular, provide learners with contextualized learning experiences that encourage creativity, autonomy, and deeper engagement, while simultaneously fostering essential skills such as planning, testing, and evaluating digital solutions.

Against this backdrop, the present study seeks to examine the impact of systematically integrating practical tasks and project-based activities into informatics lessons on students' algorithmic thinking skills. The research focuses on a group of students in a specialized school in Mingbuloq district and employs a comparative approach involving experimental and control groups. By analyzing both quantitative and qualitative learning outcomes, the study aims to determine whether practice-based instruction leads to measurable improvements in problem-solving efficiency, algorithmic accuracy, and overall learning engagement.

The study is guided by the assumption that meaningful learning in informatics emerges through active participation and continuous interaction with digital tools and problem contexts. It is hypothesized that if informatics instruction is structured around systematic practical tasks and mini-projects, students' algorithmic thinking and problem-solving abilities will improve significantly in comparison with traditional, explanation-centered teaching methods. The findings of this research are expected to contribute to the development of effective methodological recommendations for informatics teachers working in specialized secondary schools and to support the broader shift toward competence-oriented digital education.

METHODOLOGY

Participants

The study was conducted at a specialized secondary school located in the Mingbulak district. The participants consisted of sixty students enrolled in the 8th and 9th grades, selected to represent a comparable level of prior academic achievement in informatics. To ensure the reliability of the experimental findings, the students were divided into two groups of equal size. The experimental group included thirty students who participated in the practice-oriented instructional program, while the control group comprised thirty students who continued to receive instruction through conventional teaching methods. Both groups were taught by the same informatics teacher and followed the same curriculum content, which minimized the influence of external instructional variables.

Research Design

The research employed a quasi-experimental design based on a pre-test and post-test comparison of experimental and control groups. This design was selected to measure the effectiveness of the instructional intervention by identifying changes in students' algorithmic thinking and problem-solving skills over time. The duration of the experiment was six weeks, which allowed sufficient time for the systematic implementation of the practice-oriented module while maintaining consistency with the school's academic schedule. Pre-tests were administered at the beginning of the study to assess students' initial levels of algorithmic competence, and post-tests were conducted at the end of the intervention to evaluate learning outcomes.

Instructional Intervention

During the six-week period, the experimental group was taught using a structured practice-oriented instructional module designed to enhance algorithmic thinking through active engagement. Instruction emphasized the development of algorithm design skills through the use of flowcharts and pseudocode, enabling students to conceptualize problem-solving processes before coding. Learners engaged in hands-on programming tasks involving basic functions, conditional statements, and loops, which supported the gradual acquisition of core

programming concepts.

A key component of the intervention was systematic debugging practice, where students analyzed faulty code, identified logical and syntactic errors, and proposed corrected solutions. This activity aimed to strengthen analytical reasoning and reduce students' anxiety toward making mistakes. In addition, students completed mini-projects such as simple calculators, quizzes, and school-related digital applications, which required them to integrate multiple concepts and apply their knowledge in meaningful contexts. Weekly reflective activities were incorporated to promote self-assessment and metacognitive awareness, encouraging students to evaluate their learning progress, identify challenges, and set goals for improvement.

In contrast, the control group followed a traditional instructional approach characterized by teacher-centered explanations, demonstration of sample problems, and written exercises. While students in this group received the same theoretical content, their exposure to extended practical tasks and project-based activities was limited, and assessment primarily focused on written tests.

Assessment Tools

Multiple assessment instruments were used to ensure a comprehensive evaluation of students' learning outcomes. An algorithmic thinking test consisting of twenty items was administered to both groups as part of the pre-test and post-test stages. This test measured students' understanding of logical sequences, conditional reasoning, and iterative processes. Additionally, students completed two practical programming tasks that required them to design and implement functional code solutions independently. To evaluate the quality of students' mini-projects in the experimental group, a project evaluation rubric was applied, focusing on logical structure, functionality, clarity of code, and presentation quality. The combination of objective test data and performance-based assessment provided a reliable basis for comparing the effectiveness of instructional approaches.

RESULTS

The comparative analysis of pre-test and post-test data revealed substantial improvements in the performance of students in the experimental group, while progress in the control group remained moderate. The results indicate that the practice-oriented instructional intervention had a measurable impact on students' algorithmic thinking and practical programming skills.

At the beginning of the study, both groups demonstrated comparable levels of achievement across all assessed indicators, suggesting a relatively equal baseline. In the algorithmic thinking test, the experimental group achieved an average pre-test score of 56%, which increased to 78% in the post-test, representing a 22-percentage-point improvement. In contrast, the control group showed a smaller increase, rising from 55% to 63%. This difference highlights the effectiveness of systematic practical instruction in strengthening students' ability to analyze problems and design logical solutions.

A similar pattern was observed in the assessment of practical programming tasks. The proportion of students in the experimental group who correctly solved the assigned programming problems increased from 42% in the pre-test to 71% in the post-test. Meanwhile, the control group demonstrated an improvement from 41% to 52%, indicating that traditional instruction contributed to learning gains but to a lesser extent. These findings suggest that hands-on programming activities and mini-projects enabled students to apply theoretical concepts more effectively in authentic problem-solving contexts.

Significant improvement was also recorded in debugging performance. Prior to the intervention, only 38% of students in the experimental group successfully identified and corrected programming errors. Following the practice-oriented instruction, this figure rose to 69%, reflecting a marked enhancement in analytical reasoning and error-handling skills. The control group exhibited a more modest increase, from 39% to 50%, which may be attributed to limited exposure to systematic debugging exercises.

The analysis further revealed notable progress in students' use of code comments, an important indicator of algorithmic clarity and structured thinking. In the experimental group, the percentage of students who consistently used comments to

explain program logic increased from 30% to 66%. In comparison, the control group showed an increase from 31% to 44%. This result demonstrates that reflective and project-based activities encouraged students to articulate their reasoning more clearly and adopt better coding practices.

In addition to quantitative data, classroom observations provided qualitative evidence supporting these findings. Students in the experimental group displayed higher levels of engagement during lessons, actively participated in collaborative tasks, and demonstrated increased confidence when working with algorithms and code. Many students approached programming challenges more independently and showed greater persistence in resolving errors. In contrast, students in the control group tended to rely more heavily on teacher guidance and exhibited lower levels of initiative during problem-solving activities.

Indicator	Experimental Group (Pre-test)	Experimental Group (Post-test)	Control Group (Pre-test)	Control Group (Post-test)
Algorithmic test performance	56%	78%	55%	63%
Correct solution of practical tasks	42%	71%	41%	52%
Debugging success rate	38%	69%	39%	50%
Use of code comments	30%	66%	31%	44%

Overall, the results indicate that the integration of practical tasks, debugging exercises, and mini-projects into informatics instruction led to meaningful improvements in both cognitive and behavioral learning outcomes.

DISCUSSION

The findings of this study provide clear evidence that a practice-oriented approach to teaching informatics significantly enhances students' algorithmic thinking and practical programming skills in specialized school settings. The substantial gains observed in the experimental group across all assessed indicators suggest that systematic engagement with hands-on tasks and mini-projects enables learners to bridge the gap between theoretical knowledge and practical application.

One of the most notable outcomes of the intervention was the marked improvement in algorithmic test performance among students in the experimental group. This improvement indicates that exposure to structured algorithm design activities, such as flowcharts and pseudocode, strengthened students' ability to conceptualize problem-solving processes before implementation. Unlike traditional instruction, which often emphasizes memorization and procedural repetition, the practice-oriented model encouraged learners to analyze problems more deeply and develop step-by-step solutions, thereby fostering higher-order thinking skills.

The significant increase in correct solutions to practical programming tasks further underscores the effectiveness of hands-on learning. By working on real-world–oriented tasks and mini-projects, students were required to integrate multiple programming concepts simultaneously, which promoted meaningful learning and knowledge transfer. These findings align with pedagogical perspectives that emphasize active learning and project-based instruction as essential components of computational thinking development.

Improvements in debugging performance represent another critical contribution of the practice-based instructional model. Debugging tasks not only enhanced students' technical accuracy but also cultivated resilience and analytical reasoning. Regular exposure to error identification and correction appeared to reduce students' fear of making mistakes and encouraged a more experimental and reflective learning mindset. This shift is particularly important in informatics education, where trial-and-error processes are integral to professional practice.

The increased use of code comments among students

in the experimental group suggests progress in algorithmic clarity and metacognitive awareness. Commenting code requires learners to articulate their reasoning explicitly, which reinforces conceptual understanding and supports the development of structured thinking. This outcome indicates that reflective and project-based activities contribute not only to functional competence but also to the cultivation of good programming habits.

Qualitative observations complement the quantitative findings by highlighting changes in students' engagement and learning behavior. The experimental group demonstrated greater collaboration, autonomy, and confidence during programming activities. These behavioral shifts suggest that practice-oriented instruction fosters a more learner-centered classroom environment, where students take active responsibility for their learning. In contrast, the more limited progress observed in the control group may reflect the constraints of traditional, explanation-centered teaching approaches, which offer fewer opportunities for independent exploration and application.

Despite these positive outcomes, certain limitations should be acknowledged. The relatively short duration of the intervention and the restricted sample size may limit the generalizability of the findings. Future research could extend the intervention period, involve multiple schools, and incorporate longitudinal analysis to examine the long-term effects of practice-oriented informatics instruction.

Overall, the results of this study support the conclusion that integrating practical tasks, debugging activities, and mini-projects into informatics education significantly enhances students' algorithmic thinking, problem-solving abilities, and learning engagement. These findings provide a strong methodological foundation for improving informatics instruction in specialized secondary schools and contribute to ongoing discussions on effective strategies for developing computational thinking in digital education.

CONCLUSION

This study examined the impact of practice-oriented instruction on the development of algorithmic thinking and practical programming skills among students in a specialized school in the Mingbuloq district. The findings demonstrate that systematic integration of hands-on tasks, debugging exercises, and mini-projects into informatics lessons leads to significant improvements in students' cognitive and practical learning outcomes.

The comparative analysis revealed that students exposed to the practice-based instructional module outperformed their peers in the control group across all key indicators, including algorithmic test performance, correct completion of programming tasks, debugging success, and the use of structured code comments. These results indicate that practice-oriented learning environments effectively support the transformation of theoretical knowledge into functional problem-solving skills, which is a central objective of modern informatics education.

Furthermore, the study highlights the pedagogical value of learner-centered approaches that emphasize active engagement, reflection, and collaboration. By encouraging students to design algorithms, experiment with code, and analyze errors independently, the instructional model fostered greater confidence, autonomy, and motivation. Such qualities are essential for preparing learners to meet the demands of increasingly digital and technology-driven societies.

Despite its positive outcomes, the study is limited by its relatively short duration and single-school context. Future research may extend this approach to a broader range of educational settings, explore long-term learning effects, and incorporate more advanced programming concepts. Nevertheless, the results provide strong empirical support for the adoption of practice-oriented methodologies in specialized secondary schools.

In conclusion, the integration of practical tasks and project-based activities in informatics instruction represents an effective strategy for enhancing algorithmic thinking, digital competence, and overall learning quality. The findings offer valuable methodological guidance for informatics teachers seeking to improve instructional effectiveness and align classroom practice with contemporary educational goals.

REFERENCES

1. Erkulova, F. (2023). Using information and communication technologies in teaching in remote areas. Journal of Educational Technology and Innovation, 5(2), 45–52.
2. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215
3. Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books.
4. OECD. (2020). Education in the digital age: Building digital competence. OECD Publishing.
5. Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Harvard University Press.
6. UNESCO. (2021). Reimagining our futures together: A new social contract for education. UNESCO Publishing.