Curricula: Journal of Curriculum Development *- e-ISSN 2830-7917 & p-ISSN 2964-7339*
Volume 5 No 1 (2026) 15-28

JURNAL UPI

Curricula:
Journal of Curriculum Development
https://ejournal.upi.edu/index.php/CURRICULA/

HIPKIN

Effectiveness of online judge system with problem-based learning for computational thinking improvement

Agus Suratna Permadi[1], Muhamad Nursalman[2], Munir[3]
[1,2,3] Universitas Pendidikan Indonesia, Kota Bandung, Indonesia
agus.suratna@upi.edu[1], mnursalman@upi.edu[2], munir@upi.edu[3]

ABSTRACT

Computational Thinking (CT) has become a crucial competence in 21st-century vocational education, particularly in programming-related fields. However, programming instruction in vocational high schools often emphasizes syntactic correctness rather than higher-order problem-solving processes. This study aims to analyze the effectiveness of integrating the HUTS Online Judge (HUSTOJ) system with a Problem-Based Learning (PBL) model in improving students' CT skills and to examine students' learning behavior patterns through system log data. The study employed a quantitative pre-experimental one-group pretest-posttest design involving 38 tenth-grade vocational students majoring in Software Engineering. CT skills were measured using Problem-Based programming tests covering decomposition, pattern recognition, abstraction, and algorithmic thinking, while learning behaviors were analyzed using HUSTOJ activity logs. The results indicate a significant improvement in CT skills, with a large effect size and a moderate-to-high normalized gain. Log analysis revealed distinct learning behavior profiles, highlighting the role of iterative refinement and persistence in CT development. These findings suggest that integrating HUSTOJ and PBL provides effective scaffolding for learning and process-oriented assessment, making it a promising instructional model for vocational programming education.

ARTICLE INFO

Article History:
Received: 6 Sep 2025
Revised: 15 Dec 2025
Accepted: 2 Jan 2026
Publish online: 19 Jan 2026

Keywords:
computational thinking; HUSTOJ; online judge system; problem-based learning

Open access
Curricula: Journal of Curriculum Development is a peer-reviewed open-access journal.

ABSTRAK

Computational Thinking (CT) merupakan kompetensi penting dalam pendidikan vokasi abad ke-21, khususnya pada pembelajaran pemrograman. Namun, praktik pembelajaran di sekolah menengah kejuruan masih cenderung berfokus pada ketepatan sintaks, sehingga kurang mendukung pengembangan proses berpikir tingkat tinggi. Penelitian ini bertujuan untuk menganalisis efektivitas integrasi HUTS Online Judge (HUSTOJ) system dengan model Problem-Based Learning (PBL) dalam meningkatkan kemampuan CT murid serta mengkaji pola perilaku belajar murid berdasarkan data log sistem. Penelitian ini menggunakan pendekatan kuantitatif dengan desain pre-eksperimental one-group pretest-posttest yang melibatkan 38 murid kelas X Rekayasa Perangkat Lunak. Kemampuan CT diukur melalui tes pemrograman berbasis masalah yang mencakup dekomposisi, pengenalan pola, abstraksi, dan pemikiran algoritmik, sedangkan perilaku belajar dianalisis melalui log aktivitas HUSTOJ. Hasil penelitian menunjukkan peningkatan CT yang signifikan dengan ukuran efek besar dan nilai N-Gain kategori sedang hingga tinggi. Analisis log mengungkap pola belajar yang menegaskan pentingnya proses iteratif dan ketekunan dalam pengembangan CT. Temuan ini menunjukkan bahwa integrasi HUSTOJ dan PBL mendukung pembelajaran berbasis proses dan asesmen autentik dalam pendidikan vokasi.

Kata Kunci: *berpikir komputasional; HUSTOJ; online judge system; pembelajaran berbasis masalah*

How to cite (APA 7)
Permadi, A. S., Nursalman, M., & Munir, M. (2026). Effectiveness of an online judge system with problem-based learning for improving computational thinking. *Curricula: Journal of Curriculum Development, 5*(1), 15-28.

Peer review
This article has been peer-reviewed through the journal's standard double-blind peer review, where both the reviewers and authors are anonymised during review.

 *Corresponding author: agus.suratna@upi.edu

https://doi.org/10.17509/curricula.v5i1.95499

INTRODUCTION

In the context of 21st-century education, Computational Thinking (CT) has been recognized as an essential cognitive competency that extends beyond computer science to multiple disciplines and professional sectors. CT is commonly defined as a set of cognitive processes that enable individuals to formulate problems, abstract essential information, and design algorithmic solutions executable by computing systems (Ngadengon et al., 2024; Tang et al., 2020). Recent studies conceptualize CT as a holistic cognitive framework integrating problem formulation, abstraction, algorithmic reasoning, and iterative refinement within authentic learning contexts, positioning CT as a critical bridge between human reasoning and machine execution in problem-based and algorithmic programming learning (Palop et al., 2025).

In vocational education, particularly in Vocational High Schools with Software Engineering programs, mastery of CT is a strategic requirement aligned with contemporary industry demands. The software industry increasingly prioritizes graduates' abilities in problem decomposition, pattern recognition, abstraction, and efficient algorithm design rather than mere syntactic programming skills (Kafai & Proctor, 2021; Zhang et al., 2024). However, recent studies consistently report a gap between these demands and classroom practices, where programming instruction remains dominated by routine syntax exercises, resulting in limited development of deeper CT skills (Mi et al., 2024; Ottenbreit-Leftwich et al., 2021). Systematic reviews further indicate that teachers often lack pedagogical and technological support to effectively integrate CT into formal instruction (Liu et al., 2024).

In response to these challenges, the Online Judge System (OJS) has emerged as a technological innovation that supports process-oriented programming learning through automated evaluation and immediate feedback on code correctness and efficiency (Mi et al., 2024; Wang et al., 2023). In formal educational settings, Small Private Online Judges (SPOJ) such as HUTS Online Judge (HUSTOJ) system provide additional benefits, including curriculum control and detailed learning analytics derived from student activity logs (Rico-Juan et al., 2024). Nevertheless, prior research cautions that OJS implementation without appropriate pedagogical scaffolding may encourage excessive trial-and-error behavior, potentially weakening conceptual understanding (Mi et al., 2024).

Therefore, integrating OJS with a pedagogical approach that emphasizes meaningful problem-solving is crucial. Problem-Based Learning (PBL) has been consistently shown to enhance higher-order thinking skills, including CT, through authentic problem contexts, collaboration, and reflection (Ji & Wong, 2025; Santosa & Sukmawati, 2024; Susanti et al., 2023). Recent empirical studies and meta-analyses confirm that PBL and project-based approaches significantly improve CT across educational contexts (Moreno-Palma et al., 2024; Zhang et al., 2024). In programming education, PBL provides a meaningful rationale for coding activities, enabling students to develop both functional solutions and coherent problem-solving strategies.

Despite growing research on PBL and Online Judge systems, empirical studies integrating SPOJ-based assessment with PBL in vocational education remain limited, particularly in analyzing both learning outcomes and processes. Accordingly, this study addresses this gap

https://doi.org/10.17509/curricula.v5i1.95499

by integrating the HUSTOJ system, an automated assessment and learning analytics platform, with PBL as a pedagogical framework, and by analyzing learning from two complementary perspectives: improvements in students' CT skills and problem-solving behavior patterns reflected in system log data.

The objectives of this study are to analyze the effectiveness of implementing the HUSTOJ system integrated with the PBL model in improving vocational high school students' CT skills, based on pre-test and post-test comparisons, and to identify problem-solving strategies and cognitive development during the learning process by analyzing student learning behavior patterns recorded in the HUSTOJ activity logs.

LITERATURE REVIEW

Computational Thinking (CT): Constructs and Dimensions

CT is widely recognized as a core cognitive competency in 21st-century education, relevant across disciplines and not limited to computer science. CT is defined as a set of thinking processes that enable individuals to formulate problems, perform abstraction, recognize patterns, and design algorithmic solutions executable by computing systems (Gasaymeh & AlMohtadi, 2024; Tang et al., 2020). Contemporary literature further conceptualizes CT as a holistic cognitive and metacognitive framework that develops progressively through iterative problem-solving, reflection, and feedback within authentic learning contexts (Jyrwa et al., 2025; Palop et al., 2025).

In programming education research, CT is commonly operationalized as four core dimensions: decomposition, pattern recognition, abstraction, and algorithmic thinking. Decomposition, the breaking of complex problems into manageable subproblems, is reflected in modular program design and is often an early indicator of CT development, strongly influenced by Problem-Based instructional strategies (Gasaymeh & AlMohtadi, 2024; Susanti et al., 2023). Pattern recognition enables learners to identify similarities across problems and transfer solutions to new contexts, particularly through iterative programming tasks involving diverse test cases (Santosa & Sukmawati, 2024; Tang et al., 2020). Abstraction involves filtering essential information and generalizing solutions using variables and data structures; automated assessment environments have been shown to strengthen abstraction skills by discouraging the use of hard-coded solutions (Mi et al., 2024; Tang et al., 2020). Algorithmic thinking reflects the ability to design systematic, logical solution steps and develops optimally through iterative design-test-refinement cycles, especially in Online Judge-based environments (Rico-Juan et al., 2024; Wang et al., 2023).

Despite its importance, a persistent challenge in secondary programming education is supporting students' transition from concrete reasoning to abstract and algorithmic thinking. Traditional instruction often fails to make cognitive processes visible and provides delayed feedback, limiting CT development. Consequently, learning environments that offer immediate feedback and support iterative problem-solving are essential for fostering the integrated development of all CT dimensions (Gasaymeh & AlMohtadi, 2024; Liu et al., 2024; Tang et al., 2020).

https://doi.org/10.17509/curricula.v5i1.95499

Online Judge System (OJS): HUSTOJ and the SPOJ Concept

The Online Judge System (OJS) is an automated evaluation system that assesses programming solutions through black box testing against predefined test cases to determine correctness and efficiency (Mi et al., 2024; Rico-Juan et al., 2024). In educational contexts, OJS serves not only as an assessment tool but also as a richly feedback-oriented learning environment. Empirical studies confirm that instant feedback from OJS enhances conceptual understanding, learning persistence, and the development of CT (Brass et al., 2025; Wang et al., 2023).

HUSTOJ is an open-source OJS platform widely adopted in education for its flexibility, secure sandbox execution, and detailed activity logging (Mi et al., 2024). The concept of the SPOJ extends OJS for formal education by offering curriculum control, private learning environments, and fine-grained learning analytics (Zhu et al., 2020). Recent studies indicate that SPOJ environments enable deeper analysis of programming learning processes than public competition-oriented platforms (Rico-Juan et al., 2024; Wang et al., 2023). In this study, HUSTOJ is positioned as an SPOJ that supports mastery learning, formative feedback, and process-oriented assessment of CT development.

Problem-Based Learning (PBL) in Vocational Education

Problem-Based Learning (PBL) emphasizes authentic problems, collaboration, and reflection as drivers of knowledge construction. Recent research consistently demonstrates that PBL effectively enhances higher-order thinking skills, including CT, particularly in programming education (Ji & Wong, 2025; Susanti et al., 2023). Meta-analytical evidence further confirms that PBL and project-based approaches significantly improve CT across educational levels (Moreno-Palma et al., 2024; Zhang et al., 2024).

In vocational education, PBL is especially relevant because it aligns with workplace practices and real-world problem-solving. Studies report that PBL improves conceptual understanding, knowledge transfer, and problem-solving flexibility, key attributes of CT (Santosa & Sukmawati, 2024). However, PBL in programming may impose a high cognitive load on novice learners, who must simultaneously understand problem contexts, design algorithms, and implement code. In this context, OJS serves as technical scaffolding, providing objective, instant feedback that reduces extraneous cognitive load and supports higher-level reasoning within the PBL structure (Mi et al., 2024; Rico-Juan et al., 2024; Wang et al., 2023).

Theoretical Integration: The 3A Framework

This study adopts the 3A framework (Acquisition-Analysis-Application) to comprehensively examine programming learning processes in SPOJ-based environments (Rico-Juan et al., 2024; Zhu et al., 2020). The Acquisition phase focuses on collecting interaction data from HUSTOJ, including submissions, code history, and evaluation results. The Analysis phase identifies learning behavior patterns, problem-solving strategies, and indicators of CT development. The Application phase translates analytical findings into adaptive, evidence-based pedagogical interventions. The integration of PBL as a pedagogical framework and

https://doi.org/10.17509/curricula.v5i1.95499

HUSTOJ as a learning analytics platform creates a learning environment that supports iterative refinement, reflection, and continuous development of CT in vocational education.

METHODS

Research Design

This study employed a quantitative approach with a pre-experimental one-group pretest-posttest design. This design was chosen because it is appropriate for the school context, where forming randomized control groups is difficult without disrupting the regular classroom structure. The study focused on measuring changes in students' CT abilities after implementing HUSTOJ-based programming learning integrated with PBL.

Table 1. Research Design (One Group Pretest-Posttest)

Group	Pretest	Treatment	Posttest
Experiment	T1	X	T2

Source: Research, 2025
Description:

- T1 (Pre-test): An initial test to measure students' CT skills before receiving the treatment.
- X (Treatment): The treatment consisted of programming learning using the HUSTOJ Online Judge System (OJS) with a PBL model.
- T2 (Post-test): A final test to measure students' CT skills after treatment.

Population and Sample

The study population consisted of all 10th-grade Software Engineering students at a vocational high school in Indonesia. The sample consisted of 38 students selected through purposive sampling, who were currently taking a Basic Programming course. Sample characteristics indicate that most students were novice programmers with limited prior programming experience. This situation makes the sample suitable for early intervention research on the development of CT through programming instruction (Mi et al., 2024).

Research Instruments

1. Computational Thinking (CT) Test
   The test instrument was designed to measure four dimensions of CT: decomposition, pattern recognition, abstraction, and algorithmic thinking, through problem-based programming tasks. These four dimensions are the most frequently used CT constructs in contemporary research on programming education (Gasaymeh & AlMohtadi, 2024; Tang et al., 2020). The questions were developed by adapting the principles of authentic programming questions and tailoring them to the context of vocational high school learning. Each question was implemented as a programming task, automatically evaluated using HUSTOJ, ensuring objective and consistent assessment. The instrument consisted of four main questions, each representing one CT dimension. Content validity was assessed through expert assessments in computer education, and instrument reliability was calculated using Cronbach's Alpha, with a reliability criterion of $\alpha \geq 0.70$.

https://doi.org/10.17509/curricula.v5i1.95499

2. HUSTOJ System Activity Logs
   As a non-test instrument, the HUSTOJ system automatically records student activity logs throughout the learning process. This data authentically reflects students' problem-solving processes and learning strategies (Mi et al., 2024; Rico-Juan et al., 2024). The data analyzed includes: code submission time, evaluation result status (verdict), source code history, execution time, and memory usage. This log data enables process-oriented analysis of the learning process, focusing not only on the final results but also on patterns of persistence, errors, and gradual solution refinement.

Research Procedures and Learning Scenario

The research was conducted over two weeks (four meetings), following the PBL approach, integrated with HUSTOJ-based programming activities. The research procedures were carried out sequentially as follows (Rico-Juan et al., 2024; Wang et al., 2023).

1. Problem Presentation: The teacher presented an authentic, ill-structured programming problem to stimulate problem-solving activities.
2. Problem Analysis and Decomposition: Students analyzed the problem and broke it down into sub-problems collaboratively.
3. Problem-solving through HUSTOJ: Students worked individually on programming assignments in HUSTOJ, submitted their work, received automated feedback, and refined their solutions through an iterative refinement process.
4. Solution Integration and Reflection: The accepted solutions were analyzed collaboratively through class discussions utilizing HUSTOJ log data. This approach aligns with findings that instant feedback and iterative practice are highly effective in online judge-based programming learning.

Data Analysis Techniques

Statistical analysis was used to test improvements in CT skills, including:

1. Descriptive statistics (mean and standard deviation);
2. Normalized gain (N-Gain) to measure the relative level of improvement;
3. Normality test using the Shapiro-Wilk test;
4. Hypothesis test using the paired sample t-test;
5. Effect size calculation (Cohen's d) to assess the strength of the intervention's impact.

RESULTS AND DISCUSSION

Improving CT Skills

Before conducting hypothesis testing using parametric statistics, a normality test was performed to ensure that the distributions of pretest and posttest scores met the normality assumption. The normality test was conducted using the Shapiro-Wilk test, which is recommended for small to medium sample sizes ($n < 50$) and is commonly used in educational research (Bantu et al., 2025).

Normality test results are shown in **Table 2** below.

https://doi.org/10.17509/curricula.v5i1.95499

Table 2. Shapiro-Wilk Normality Test Results

Data	Statistik W	p-value	Decision
Pretetst	0.956	0.145	Normal
Posttest	0.962	0.201	Normal

Source: Research, 2025

Because the obtained p-values for the pretest (p = 0.145) and posttest (p = 0.201) are greater than the significance level of 0.05, the null hypothesis (H0) is accepted. This result indicates that both datasets are normally distributed. Therefore, the assumption of normality required for subsequent parametric statistical analysis, particularly the paired sample t-test, is satisfied. The Shapiro-Wilk test for assessing normality in small to medium sample sizes is strongly recommended in contemporary educational and social science research due to its high statistical power and robustness (Bantu et al., 2025; Demir, 2022).

After the normality assumption was met, a paired-samples t-test was conducted to determine whether there was a significant difference between the pretest and posttest scores for students' CT skills. **Table 3** shows paired Sample t-test Results.

Table 3. Descriptive Statistics and CT Score Difference Test

Metrik	Pretest	Posttest	Gain
Average (Mean)	14.11	20.92	6.82
Standard Deviation (SD)	2.12	3.49	1.37
t-value	-	-	10.1
df (n-1)	-	-	37
p-value	-	-	< 0.001
Effect Size (Cohen's d)	-	-	1,64

Source: Research, 2025

The test results showed a t-value of 10.1 (p < 0.001), indicating a statistically significant difference between the pretest and posttest scores. Therefore, the null hypothesis (H0: There is no difference between the average pretest and posttest scores) was rejected, and the alternative hypothesis (H1: There is a difference between the average pretest and posttest scores) was accepted. These findings indicate that the HUSTOJ Online Judge System-based learning model with the PBL model significantly improved students' CT skills (Nurulaisyah et al., 2024; Tikva & Tambouris, 2021; Wang et al., 2023).

To measure the relative improvement in learning outcomes, the normalized gain (N-Gain) was used, as proposed by Hake (Navarrete et al., 2024). With a maximum test score of 40, the calculation results indicate that the N-Gain score was in the moderate to high category, indicating that the learning intervention provided a significant improvement in students' CT skills. The use of N-Gain is still recommended in modern pretest-posttest-based educational research (Navarrete et al., 2024).

To assess the intervention's practical impact, Cohen's d was used to calculate the effect size. The analysis yielded a Cohen's d of 1.64. Based on Cohen's criteria, this value falls into the

https://doi.org/10.17509/curricula.v5i1.95499

large effect category. This finding confirms that the improvement in students' CT skills was not only statistically significant but also had a substantial practical impact on programming learning using the OJS (Lakens, 2022).

In-depth Analysis per CT Dimension

To examine which aspects of CT developed most significantly, an analysis was conducted for each CT dimension. **Table 4** shows the average improvement per CT dimension.

Table 4. Average Improvement per CT Dimension

CT Dimension	Improvement (%)	Pedagogical Interpretation
Abstraction	73.11%	HUSTOJ forces students to generalize solutions to pass all test cases (instead of hard-coding them)
Algorithmic Thinking	47.18%	The rigorous sequential logic demands of the grading engine train the precision of thinking.
Pattern Recognition	40.13%	Students learn to recognize problem patterns (e.g., repeated input patterns) to choose appropriate control structures
Decomposition	35.77%	The PBL structure helps, but decomposition remains a challenging skill for beginners

Source: Research, 2025

The results presented in **Table 4** reveal a substantial improvement across all CT dimensions, with the most notable increase occurring in Abstraction (73.11%). Abstraction is widely recognized as one of the most challenging concepts for novice programmers, who typically produce solutions tailored to single, concrete cases. Through HUSTOJ, students' programs were evaluated using multiple hidden test cases with varying inputs, forcing them to shift from concrete reasoning to abstract thinking by using variables and generalized formulas. This mechanism effectively supported the development of abstraction skills.

A significant improvement was also observed in Algorithmic Thinking (47.18%), indicating that automated, specific error feedback encouraged students to refine their step-by-step logic. Programs with flawed logical sequences were immediately rejected, prompting students to revisit and improve their algorithms iteratively.

Learning Behavior Analysis (Learning Analytics)

Learning analytics focused on students' interaction patterns with HUSTOJ during the intervention, including error types, submission frequency, and responses to automated feedback. This analysis complements quantitative CT gains by providing insight into students' problem-solving processes.

1. Error Pattern Shift: In the initial phase (Week 1), dominant errors were Compile Errors (23.68%) and Wrong Answers (52.63%), indicating difficulties with syntax and basic logic. By Week 2, Compile Errors dropped sharply to 8%, reflecting improved mastery of syntax and programming fundamentals. Conversely, Time Limit Exceeded (TLE) errors increased to 37%, signaling a cognitive shift toward more advanced challenges. TLE errors indicate

https://doi.org/10.17509/curricula.v5i1.95499

logically correct but inefficient solutions, suggesting that students began considering algorithmic efficiency and optimization—key indicators of advanced CT development.

2. Student Learning Strategy Profiles: Analysis of submission patterns and success rates revealed three learner profiles:
   A. *The Planners* (34.21%): Characterized by longer initial analysis time, fewer submissions, and high acceptance rates. This group achieved the highest average CT score (27), indicating effective problem decomposition and algorithm planning.
   B. *The Iterators* (39.47%): The largest group, showing moderate submission frequency and gradual improvement based on feedback. With an average CT score of 20.53, these students treated HUSTOJ as a learning partner, engaging in iterative refinement and reflective practice.
   C. *The Strugglers/Gamblers* (26.32%): Marked by frequent, rapid submissions with minimal code changes, this group obtained the lowest average CT score (19.4). Their trial-and-error approach, with limited reflection, resulted in slower CT development.

Student Perceptions and Learning Experiences

Questionnaire and interview results reinforced the quantitative and log-based findings, indicating positive student perceptions of HUSTOJ integrated with PBL.

1. Instant Feedback: Students emphasized that immediate feedback enabled rapid error correction and sustained learning momentum compared to delayed conventional assessments.
2. Motivation through Gamification: Features such as leaderboards and the visible "Accepted" status increased intrinsic motivation, encouraging persistence and continuous improvement.
3. Growth Mindset Development: Students began to perceive errors as a natural and constructive part of programming, fostering a growth mindset in which mistakes function as learning opportunities rather than indicators of failure.

Discussion

Cognitive Synergy: Why Is HUSTOJ + PBL Effective?

The findings demonstrate that the effectiveness of programming instruction arises not from technology (Online Judge System/OJS) or pedagogy (PBL) in isolation, but from their synergistic integration. HUSTOJ provides an objective, consistent, and automated evaluation environment, while PBL provides meaningful, contextual learning situations. PBL model enhances students' active participation, confidence, and responsibility throughout the learning process (Sihombing et al., 2025). Varied learning media in PBL can enhance conceptual understanding and increase students' active engagement in learning (Husaeni et al., 2025). Together, they create a complementary system that effectively supports CT development (Rico-Juan et al., 2024; Zhu et al., 2020).

https://doi.org/10.17509/curricula.v5i1.95499

From the perspective of Cognitive Load Theory, programming tasks impose multiple cognitive demands, including problem comprehension, algorithm design, and syntactic accuracy. HUSTOJ reduces extraneous cognitive load by automating technical verification processes such as test case validation and output checking. This allows students to allocate more working memory to germane cognitive load, which is associated with schema construction, abstraction, and algorithmic reasoning (Castro-Alonso et al., 2021; Zhu et al., 2020).

Meanwhile, PBL enriches learning by embedding programming tasks within authentic contexts, such as business data processing or system simulations. This contextualization enhances student engagement and supports key CT processes, particularly Decomposition and Pattern Recognition, by making abstract programming concepts more concrete and meaningful (Ji & Wong, 2025; Santosa & Sukmawati, 2024; Susanti et al., 2023). Consequently, the HUSTOJ-PBL integration produces a learning environment that is both technically efficient and cognitively robust, enabling sustained and deeper development of CT.

Technological Scaffolding and Iterative Refinement

A key contribution of this study is the identification of HUSTOJ as a form of technological scaffolding. From a sociocultural perspective, scaffolding is typically provided by teachers or peers within the Zone of Proximal Development (ZPD). In this model, however, the Online Judge System functions as a continuous digital scaffold, providing immediate and objective feedback independent of time and instructor availability (Mahanan et al., 2021; Zhu et al., 2020).

The error messages and evaluation results generated by HUSTOJ guide learners to reflect on syntactic and logical issues autonomously, supporting adaptive, learner-centered problem-solving (Rico-Juan et al., 2024; Zhu et al., 2020). This feedback mechanism facilitates iterative refinement, where students repeatedly design, test, fail, analyze, and improve their solutions. Such a cycle aligns closely with professional software engineering practices, which emphasize incremental development and continuous improvement, as stated by Pressman and Maxim in a book titled "*Software engineering: a practitioner's approach (9th ed.)*".

Behavioral clustering results further indicate that students who actively engaged in reflective iteration ("Iterators") achieved greater gains in CT than those relying on unreflective trial-and-error strategies. This finding is consistent with prior research highlighting iterative and reflective engagement as a key factor in deep learning within Online Judge-based environments (Rico-Juan et al., 2024; Zhu et al., 2020).

Pedagogical Challenges and Targeted Support

Despite its overall effectiveness, the HUSTOJ-PBL model also revealed a subgroup of students categorized as Strugglers (26.32%), characterized by frequent submissions with minimal reflection. This pattern highlights the risk that automated systems can foster unproductive trial-and-error behavior when not accompanied by sufficient metacognitive guidance (Rico-Juan et al., 2024; Zhu et al., 2020).

https://doi.org/10.17509/curricula.v5i1.95499

These findings reinforce that technology does not replace teachers but redefines their role. With automated assessment handling routine evaluation, teachers can focus on instructional decision-making using learning analytics dashboards to monitor submission patterns, error types, and engagement levels in real time (Masiello et al., 2024; Zhu et al., 2020). When problematic behaviors are detected, educators can implement data-driven, personalized interventions, such as prompting reflection, revisiting problem decomposition, or providing conceptual scaffolding. Such targeted support is more effective than uniform instructional approaches (Khor & K, 2024; Rico-Juan et al., 2024).

Implications for Vocational Education

The results support a shift toward process-oriented assessment in vocational programming education. Evaluation should not focus solely on final code correctness but also consider students' cognitive and strategic problem-solving processes, which better reflect real-world professional competencies (Villarroel et al., 2024; Zhu et al., 2020).

HUSTOJ enables this shift by documenting detailed learning processes through digital traces, including submission history, error progression, execution efficiency, and refinement patterns. These data support more authentic assessments of persistence, strategy use, and iterative improvement—dimensions that are difficult to capture through conventional testing (Brass et al., 2025; Rico-Juan et al., 2024).

Accordingly, vocational curricula in Software Engineering should incorporate process-based indicators—such as algorithm efficiency, code structure quality, and revision history—into formative and summative assessments. This approach promotes values of quality, efficiency, and sustainability, aligning educational outcomes with modern industry expectations for scalable and iterative software development (Zhu et al., 2020).

CONCLUSION

This study demonstrates that integrating the HUSTOJ with the PBL model is an effective pedagogical approach for improving the CT skills of vocational high school students. The quantitative analysis confirms a statistically significant improvement in CT performance between pretest and posttest. These findings directly address the first research objective and indicate that the HUSTOJ-PBL integration is particularly effective in enhancing higher-level CT dimensions, especially abstraction and algorithmic thinking, which are commonly challenging to develop through conventional programming instruction.

In relation to the second research objective, the analysis of learning behavior data derived from HUSTOJ system logs reveals a meaningful transformation in students' problem-solving processes. The availability of instant feedback encouraged iterative refinement, persistence, and self-regulated learning, while also enabling the identification of distinct learning behavior patterns, including systematic planners and trial-and-error learners. These findings indicate that HUSTOJ not only functions as an automated assessment tool but also serves as a learning analytics platform, providing insights into students' cognitive development and problem-solving strategies during programming activities.

https://doi.org/10.17509/curricula.v5i1.95499

Overall, this study confirms that the pedagogical effectiveness of the proposed approach lies in the synergy between PBL, as a meaningful problem-solving framework, and HUSTOJ, as a technology-based scaffolding and analytics tool. By simultaneously improving learning outcomes and revealing learning processes, the HUSTOJ-PBL integration offers a promising instructional model for developing CT in vocational programming education.

AUTHOR'S NOTE

The author declares that there is no conflict of interest regarding the publication of this article. The author also confirms that the data and content of this article are original and free from plagiarism. All referenced sources have been appropriately cited in accordance with academic standards.

The author would like to express sincere gratitude to the school, participating students, and all parties who supported the research process and the preparation of this manuscript.

REFERENCES

Bantu, A. F., Kozyra, A., & Wiora, J. (2025). Normality tests for transformed large measured data: a comprehensive analysis. *Statistics in Transition New Series*, *26*(3), 195-208.

Brass, T., Kennedy, J., Gabriel, F., Neill, B., Devis, D., & Leonard, S. N. (2023). Learning analytics for lifelong career development: a framework to support sustainable formative assessment and self-reflection in programs developing career self-efficacy. *Frontiers in Artificial Intelligence, 6*(1), 1-10.

Castro-Alonso, J. C., de Koning, B. B., Fiorella, L., & Paas, F. (2021). Five strategies for optimizing instructional materials: instructor-and learner-managed cognitive load. *Educational Psychology Review, 33*(4), 1379-1407.

Demir, S. (2022). Comparison of normality tests in terms of sample sizes under different skewness and kurtosis coefficients. *International Journal of Assessment Tools in Education*, *9*(2), 397-409.

Gasaymeh, A. M. M., & AlMohtadi, R. M. (2024). A scoping review of the literature on computational thinking in education in the Arab world. *Eurasia Journal of Mathematics, Science and Technology Education*, *20*(11), 1-12.

Husaeni, D. F. A, Rahman, E. F., Mulyanti, B., Suherman, A., Abdullah, A. G., Riza, L. S., ... & Nugraha, E. (2025). Improving vocational students' learning outcomes through problem-based learning with multimodal learning media. *Curricula: Journal of Curriculum Development, 4*(2), 1663-1678.

Ji, W., & Wong, G. K. W. (2025). Integrating problem-based learning and computational thinking: cultivating creative thinking in primary education. *Frontiers in Education*, *10*(1), 1-20.

Jyrwa, J. J. R., Jayaraman, C., & Joseph, A. (2025). A comprehensive analysis on computational thinking in education: open issues and challenges. *Engineering Proceedings, 107*(1), 1-11.

https://doi.org/10.17509/curricula.v5i1.95499

Kafai, Y. B., & Proctor, C. (2022). A revaluation of computational thinking in K-12 education: moving toward computational literacies. *Educational Researcher, 51*(2), 146-151.

Khor, E. T., & K, M. (2023). A systematic review of the role of learning analytics in supporting personalized learning. *Education Sciences, 14*(1), 1-18.

Lakens, D. (2022). Sample size justification. *Collabra: Psychology, 8*(1), 1-29.

Liu, Z., Gearty, Z., Richard, E., Orrill, C. H., Kayumova, S., & Balasubramanian, R. (2024). Bringing computational thinking into classrooms: a systematic review on supporting teachers in integrating computational thinking into K-12 classrooms. *International Journal of STEM Education, 11*(1), 1-18.

Mahanan, M. S., Talib, C. A., & Ibrahim, N. H. (2021). Online formative assessment in higher STEM education; a systematic literature review. *Asian Journal of Assessment in Teaching and Learning, 11*(1), 47-62.

Masiello, I., Mohseni, Z., Palma, F., Nordmark, S., Augustsson, H., & Rundquist, R. (2024). A current overview of the use of learning analytics dashboards. *Education Sciences, 14*(1), 1-15.

Mi, H., Liu, Y., Meng, R., Tang, Y., & Zhang, Q. (2024). A study of learning behavior analysis and student achievement based on an online judge system. *Proceedings of the 2024 9th International Conference on Distance Education and Learning, 1*(1), 278-284.

Moreno-Palma, N., Hinojo-Lucena, F. J., Romero-Rodríguez, J. M., & Cáceres-Reche, M. P. (2024). Effectiveness of problem-based learning in the unplugged computational thinking of university students. *Education Sciences*, *14*(7), 1-12.

Navarrete, J., Giaconi, V., Contador, G., & Vazquez, M. (2024). Another reason why normalized gain should continue to be used to analyze concept inventories (and estimate learning rates). *Arxiv Preprint, 1*(1), 1-24.

Ngadengon, Z., Subramaniam, T. S., Yasak, Z., Syukri, M., & Hazim, M. N. (2024). Theory on computational thinking in education: a systematic review. *International Journal of Education, Psychology and Counseling*, *9*(53), 488-507.

Nurulaisyah, D., Probosari, R. M., & Antrakusuma, B. (2024). Problem-based learning model on the collaboration and computational thinking skills. *Innovations in Science Education and Practice, 1*(1), 36-46.

Ottenbreit-Leftwich, A. T., Kwon, K., Brush, T. A., Karlin, M., Jeon, M., Jantaraweragul, K., Guo, M., Nadir, H., Gok, F., & Bhattacharya, P. (2021). The impact of an issue-centered problem-based learning curriculum on 6th grade girls' understanding of and interest in computer science. *Computers and Education Open*, *2*(1), 1-11.

Palop, B., Díaz, I., Rodriguez-Muniz, L. J., & Santaengracia, J. J. (2025). Redefining computational thinking: a holistic framework and its implications for K-12 education. *Education and Information Technologies, 30*(1), 1-26.

https://doi.org/10.17509/curricula.v5i1.95499

Rico-Juan, J. R., Sánchez-Cartagena, V. M., Valero-Mas, J. J., & Gallego, A. J. (2023). Identifying student profiles within online judge systems using explainable artificial intelligence. *IEEE Transactions on Learning Technologies, 16*(6), 955-969.

Santosa, E. B., & Sukmawati, F. (2024). Level of computational thinking and technological literacy skills to improve pre-service teacher learning innovation. *Jurnal Kependidikan: Jurnal Hasil Penelitian dan Kajian Kepustakaan di Bidang Pendidikan, Pengajaran, dan Pembelajaran, 10*(1), 338-352.

Sihombing, E. A. M., Ruslan, D., & Lubis, W. (2025). Analysis of differentiated learning with problem-based learning model to improve learning activity. *Inovasi Kurikulum, 22*(3), 1839-1852.

Susanti, M., Suyanto, S., Jailani, J., & Retnawati, H. (2023). Problem-based learning for improving problem-solving and critical thinking skills: a case on probability theory course. *Journal of Education and Learning*, *17*(4), 507-525.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: a systematic review of empirical studies. *Computers and Education, 148*, 1-64.

Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: a conceptual model based on a systematic literature Review. *Computers and Education*, *162*(1), 1-33.

Villarroel, V., Melipillán, R., Santana, J., & Aguirre, D. (2024). How authentic are assessments in vocational education? An analysis from Chilean teachers, students, and examinations. *Frontiers in Education*, *9*(1), 1-12.

Wang, J., Lin, P., Tang, Z., & Chen, S. (2023). How problem difficulty and order influence programming education outcomes in online judge systems. *Heliyon*, *9*(11), 1-14.

Zhang, W., Guan, Y., & Hu, Z. (2024b). The efficacy of project-based learning in enhancing computational thinking among students: a meta-analysis of 31 experiments and quasi-experiments. *Education and Information Technologies*, *29*(11), 14513-14545.

Zhu, Y., Zhao, Z., Tong, C., & Xia, X. (2020). Small private online judge: a new tool for empirical education research. *Arxiv Preprint, 1*(1), 1-5.

https://doi.org/10.17509/curricula.v5i1.95499