SCORE PREDICTION FROM PROGRAMMING EXERCISE SYSTEM LOGS USING MACHINE LEARNING

Tetsuo Tanaka[1] and Mari Ueda[2]
[1]*Dept. of Information and Computer Sciences, Faculty of Information Technology, Kanagawa Institute of Technology*
1030 Shimo-ogino, Atsugi, Kanagawa, 243-0292, Japan
[2]*Dept. of Information Media, Faculty of Information Technology, Kanagawa Institute of Technology*
1030 Shimo-ogino, Atsugi, Kanagawa, 243-0292, Japan

ABSTRACT

In this study, the authors have developed a web-based programming exercise system currently implemented in classrooms. This system not only provides students with a web-based programming environment but also tracks the time spent on exercises, logging operations such as program editing, building, execution, and testing. Additionally, it records their results. For educators, the system offers insights into each student's progress, the evolution of their source code, and the instances of errors. While teachers find these functions beneficial, the method of providing feedback to students needs improvement. Immediate feedback is proven to be more effective for student learning. If the final course score could be predicted based on early data (e.g., from the 1st or 2nd week), students could adapt their study strategies accordingly. This paper demonstrates that one can predict the final score using the system's operational logs from the initial phases of the course. Furthermore, the score predictions can be revised weekly based on new class logs. We also explore the potential of offering tailored advice to students to enhance their final score.

KEYWORDS

Programming Exercise, Score Prediction, Effective Feedback, Operation Log, Log Analysis

1. INTRODUCTION

The digital learning environment is expanding. Within programming education, services such as Google Colaboratory, paiza.io, and Replit have emerged, which enable students to program directly in browsers without constructing personal programming environments. On these digital platforms, student activity logs and learning histories are automatically recorded. Research utilizing these datasets to pinpoint areas where students struggle has been conducted [Ohno 2022]. Furthermore, dashboards have been designed to consolidate various learning metrics, offering valuable insights to both educators and learners [Kia 2020; Sedrakyan 2019; Susnjak 2022; Khulbe 2023]. Such dashboards have evolved into essential feedback tools [Raubenheimer 2021] that substantially influence academic outcomes.

However, a challenge persists: programming courses often report high failure rates. Early performance prediction has been proposed as a remedy for the same [Carter 2019; Quille 2019; Liu 2023; Sobral 2021]. Score prediction stands out as a potent feedback tool. Early identification of underperforming students enables educators to intervene with timely advice. This anticipatory feedback allows students to understand their current academic standing, adjust their study schedules, and even redefine their learning goals. Such feedback is pivotal for strategic study planning.

We have designed and implemented a programming practice system used in real-world classrooms [Satoh 2022; Tanaka 2023a]. Presently, our system provides educators with insights into overall class trends and individual student exercise statuses via a dashboard. However, the student dashboard remains limited, and its feedback capabilities are not yet fully realized. We aim to augment the system with more effective feedback mechanisms. Our prior work [Tanaka 2023b] delved into preliminary analyses of programming exercise logs to enhance student feedback. This paper delves deeper, focusing on predicting final exam scores based on those preliminary findings.

ISBN: 978-989-8704-52-8 © 2023

Immediate feedback has emerged as most effective for students. If one can predict a course's final score based on early-stage data (e.g., from the 1st and 2nd weeks of a term), students can recalibrate their learning approaches accordingly. Conversely, if students receive unfavorable predictions early on, it might demotivate them, causing them to resign prematurely. To circumvent such setbacks, it is crucial to ensure that the students remain motivated. They should understand that even if initial projections are unsatisfactory, dedicated efforts can reverse the tide. We demonstrate that the final score can be predicted using early course system logs and that these predictions can be regularly revised as new weekly logs are generated. Additionally, we explore strategies for tailored advice to improve final scores.

Moving forward, Section 2 delves into the system's architecture, while Section 3 elaborates on the methodologies and outcomes of score prediction.

2. PROGRAMMING EXERCISE SYSTEM OVERVIEW

As depicted in Figure 1, the developed programming exercise system offers students an online programming environment and provides instructors with insights into their coding status [Satoh 2022; Tanaka 2023a]. Whenever a student interacts with a programming environment—be it through keyboard inputs, clicks on execution buttons, or any other actions—the system logs various details on a server. This includes the specific time of each action and the content displayed within the editor, console, standard output, and output files at that instance. The system formats the log and presents a list of practice situations for each student to the teacher. Currently, the system exclusively supports the C programming language.

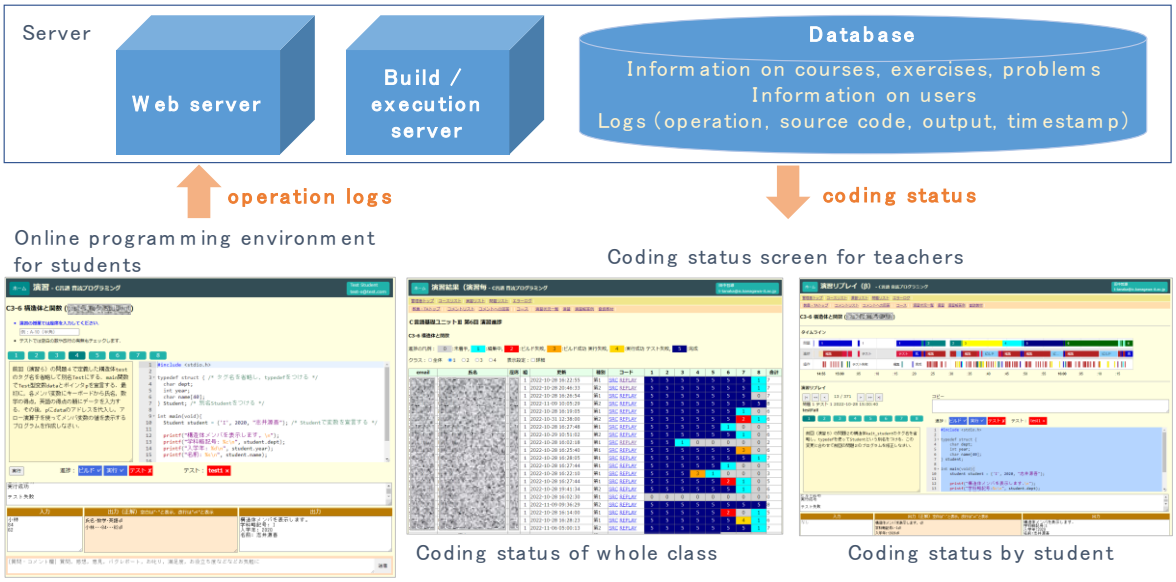


Figure 1. Programming exercise system overview

The logged data comprise various details, as listed below.

- Exercise information: course ID, exercise ID, problem ID
- User details: user ID, email, name
- Operational data and outcomes, which include the following.
  - Operation type (e.g., display, problem switching, blur, edit, copy and paste, build/execute/test)
  - Coding status indicators (e.g., not started, editing, build failed, execution failed, test failed, test successful, specific test cases passed or failed)
- Contents of the editor, console, standard output and error outputs, output files, and timestamps of each operation.

On the educator's dashboard, as illustrated in Figure 1, a comprehensive list comprising the coding statuses of the entire class is displayed. It provides details such as user ID, username, seat number, class, last update timestamp, and the progression status of each problem being worked on. The progression is categorized as *not started*, *editing*, *build failed*, *build succeeded (but execution failed)*, *execution succeeded (but test failed)*, and *completed*. Furthermore, hyperlinks to the latest snapshot of a student's practice situation and a link showcasing a chronological animation of their practice history are also available. This layout aids teachers in identifying students who might be struggling, even if they have not explicitly sought help.

Additionally, the system offers instructors a real-time glimpse into each student's coding scenario. On this screen, educators can directly execute the student's code and inspect the outcomes. This feature enables teachers to review the specific code written by students and diagnose the types of errors that might be appearing on their respective devices, be it PCs or tablets. Another highlight of the system is its ability to present a dynamic representation of a student's coding journey. Here, the system portrays an animated timeline, detailing the problems tackled, their progression stages (like editing, executing, testing), and a chronological record of user interactions and results.

3. SCORE PREDICTION

To utilize the operation logs of the programming practice system for effective feedback to students, we conducted a preliminary log analysis. This analysis confirmed that variables such as the total working time, the total number of program executions, and the total number of completed programs influenced the prediction results [Tanaka 2023b].

Providing feedback to students immediately is most effective. Predicting the final course score based on early data (from the 1st or 2nd week) can empower students to adjust their learning habits and study plans as required. However, there is a potential downside: predictions of poor scores at an early stage can demotivate students and lead to them giving up. To prevent such a scenario, the students must be reassured that, even with a less-than-ideal initial prediction, they can improve with continued effort and further learning.

In this study, our goal is to predict the final score on a weekly basis by using the accumulated data, and thus provide weekly feedback to students. Our initial approach involves creating a multiple regression analysis model using data from actual classes conducted in the latter half of the 2022 academic year.

3.1 Analysis Target

During the 3rd and 4th quarters (spanning seven weeks each) of 2022, our system was employed in the "C language unit" classes at the Department of Information and Computer Science, Faculty of Informatics, Kanagawa Institute of Technology, Japan. Here are the specifics.

- The system served approximately 230 students, predominantly first-years, in a 90-min seminar class.
- The units included C language unit II (one class), C language unit III (four classes), and C language unit IV (one class).
- Examples of answers are published after each class.
- The class sizes varied, ranging from 10 to 45 students
- In a single 90-minute class, each student generated 250 to 300 logs. This count doubled to 500 to 600 logs when considering independent study post-class.

For the analysis, our data targets were set as follows.

- Course: C language unit III, which has the highest student enrollment.
- Log acquisition period: 3rd quarter of 2022.
- Student Cohort: 145 first-year students who enrolled in C language unit II in the 2nd quarter and unit III in the 3rd quarter.
- Log Types Analyzed: We considered various parameters, including the score from C language unit II, time spent, number of execution attempts, number of problems tackled, and number of programs completed. These were aggregated both during class and post-class.

3.2 Prediction Method

In the 3rd and 4th quarters of 2022, we implemented the system in real-world classes. From this data, we developed a multiple regression analysis model with the following parameters.

- The objective variable is defined as the final examination scores from the C Language Unit III in the 3rd quarter of 2022.
- Explanatory variables comprise the final examination scores from C Language Unit II in the 2nd quarter of 2022 and five types of logs from the 3rd quarter. These logs include the time spent, number of trials, number of attempted problems, number of completed problems, and the average time to completion, both during class and post-class.
- We used Python and Scikit-learn for analysis. The explanatory variables were standardized using sklearn.preprocessing.StandardScaler, and then trained using sklearn.linear_model.LinearRegression.
- Predictions began based solely on the scores from C Language Unit II from the previous quarter. We then added logs incrementally from each successive class, from the first to the seventh.
- Model evaluation metrics include the mean squared error (MSE), the coefficient of determination (R^2), p-value Prob(F), adjusted coefficient of determination Adj.-R2, and partial regression coefficient for training data and test data.

3.3 Prediction Results

Table 1 presents the prediction results. Results (1)–(8) detail predictions starting solely with the scores of C Language Unit II followed by results obtained after adding logs from each class, up to the seventh class. The table data are used to generate a scatterplot of the predicted vs. actual scores, the MSE of the training data, R^2, the p-value (Prob(F)) of the test, adjusted R^2 (Adj.- R2), and the MSE and R^2 for the test data.

The Prob(F) statistic, derived from the training data for each model, is notably low, highlighting the utility of each model. Additionally, the adjusted R^2 for each model exceeds 0.5, suggesting good accuracy—though it is neither stellar nor poor. Conversely, as more class data gets incorporated into predictions, the accuracy of the model augments for the training data but declines for the test data, evidenced by the increasing MSE and decreasing R^2. Notably, post the fifth model—which incorporates 4th lesson logs—R^2 falls below 0.4. The scatterplots reflect this trend, showing test data predictions significantly diverging from actual values and thus indicating overfitting.

Table 2 lists partial regression coefficients for model (2), grounded on C2 scores and the logs of the first class. Only two variables, C2 score and the number of attempted problems during class, registered p-values under 0.05. Given the observed correlation among variables, addressing multicollinearity is imperative; this will be addressed in future research.

These findings affirm the feasibility of early-stage predictions and feedback regarding students' final exam scores, especially at the course's outset (the 1st and 2nd weeks), despite extant challenges.

Table 1. Prediction results

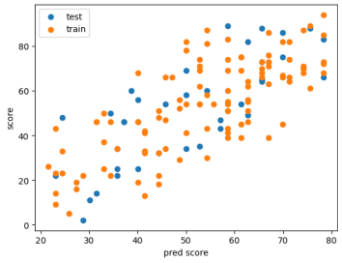
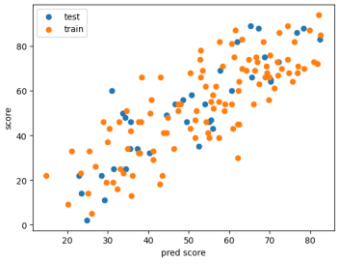
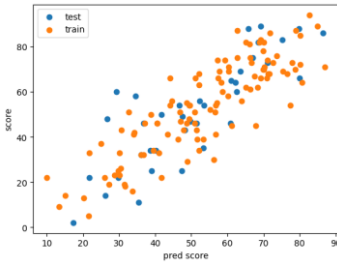
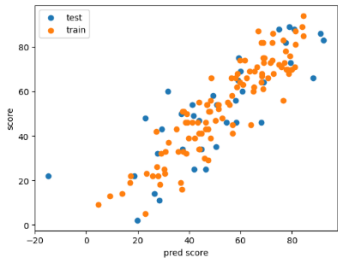
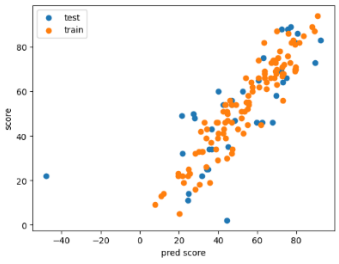
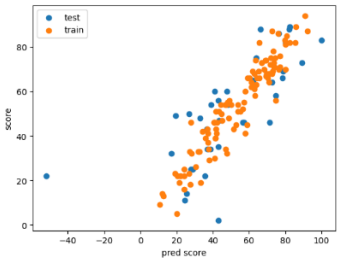
	(1) C2 score only	(2) C2 score, logs of 1st lesson	(3) C2 score, logs of 1st–2nd lessons
Scatter plot			
Training	MSE: 188.5 R^2: 0.561 Prob(F): 1.2e-20 Adj. R^2: 0.556	MSE: 151.9 R^2: 0.646 Prob(F): 3.0e-17 Adj. R^2: 0.605	MSE: 114.2 R^2: 0.734 Prob(F): 9.4e-17 Adj. R^2: 0.669
Test	MSE: 213.3 R^2: 0.592	MSE: 142.0 R^2: 0.731	MSE: 165.4 R^2: 0.684

Table 1. Prediction results (continued)

	(4) C2 score, logs of 1st–3rd lessons	(5) C2 score, logs of 1st–4th lessons	(6) C2 score, logs of 1st–5th lessons
Scatter plot			
Training	MSE: 74.0 R^2: 0.828 Prob(F): 2.30e-18 Adj. R^2: 0.757	MSE: 52.4 R^2: 0.878 Prob(F): 7.8e-18 Adj. R^2: 0.802	MSE: 44.8 R^2: 0.896 Prob(F): 1.6e-14 Adj. R^2: 0.800
Test	MSE: 201.9 R^2: 0.615	MSE: 331.4 R^2: 0.367	MSE: 362.2 R^2: 0.309

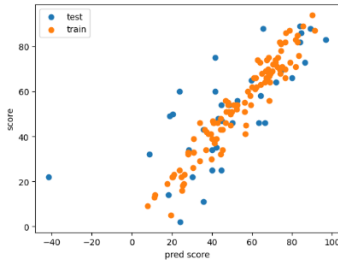
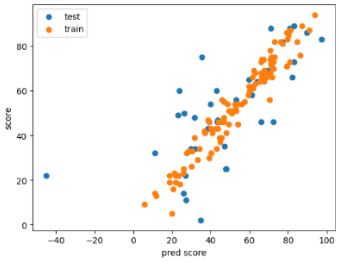
	(7) C2 score, logs of 1st–6th lessons	(8) C2 score, logs of 1st–7th lessons
Scatter plot		
Training	MSE: 31.6 R^2: 0.926 Prob(F): 4.8e-13 Adj. R^2: 0.829	MSE: 24.4 R^2: 0.943 Prob(F): 3.2e-10 Adj. R^2: 0.830
Test	MSE: 357.2 R^2: 0.318	MSE: 395.5 R^2: 0.245

Table 2. Summary of model (2), which is based on the C2 score and the logs of the 1st class

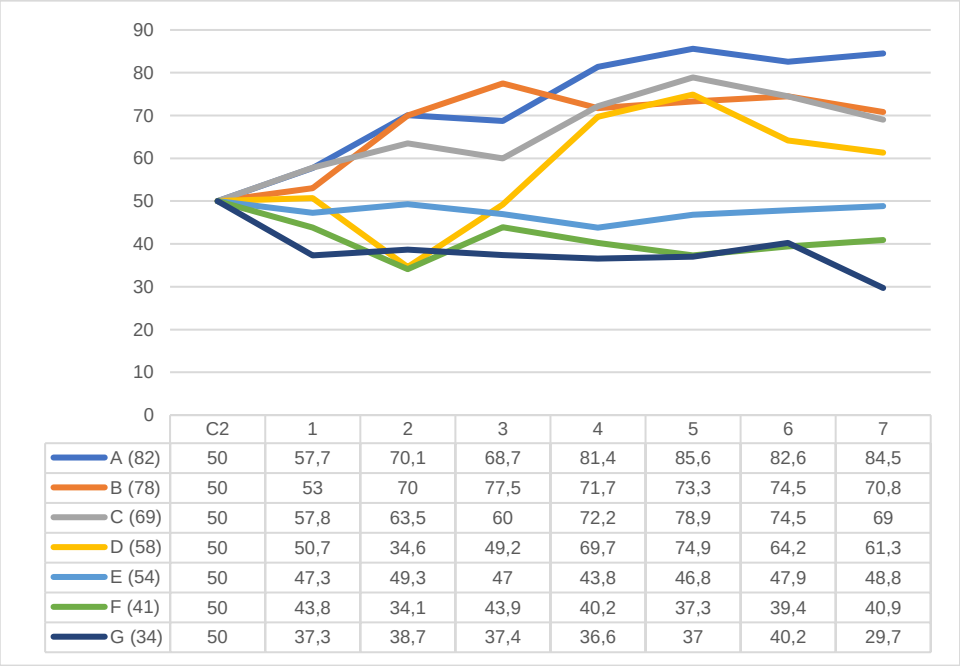
	coef	std err	t	P > |t|	[0.025	0.975]
Const	53.4722	1.258	42.506	0	50.975	55.969
C2 score	12.7118	1.475	8.619	0	9.784	15.639
During class						
Time spent	1.1715	2.64	0.444	0.658	−4.069	6.412
Number of trials	−4.1953	2.613	−1.606	0.112	−9.382	0.991
Number of attempted problems	7.0018	3.086	2.269	0.025	0.877	13.127
Number of completed problems	−1.0812	3.01	−0.359	0.720	−7.057	4.895
Average completion time	0.8762	1.596	0.549	0.584	−2.292	4.044
During and after class						
Time taken	0.4645	3.882	0.120	0.905	−7.242	8.171
Number of trials	2.2258	2.705	0.823	0.413	−3.144	7.595
Number of tried problems	3.7181	5.040	0.738	0.462	−6.286	13.722
Number of completed problems	−5.2563	4.989	−1.054	0.295	−15.159	4.646
Average of completion time	−4.2947	3.852	−1.115	0.268	−11.94	3.351

3.4 Changes in the Predicted Scores of Students with the Same Initial Predictions

An initial low prediction based on the previous quarter's C2 scores could demotivate students or lead them to consider dropping the subject. To counter this potential discouragement, demonstrating that, even if the early-stage predicted score is low, significant improvement can be achieved through subsequent efforts is crucial.

To this end, Figure 2 displays the evolving predicted scores of seven students (A–G) who all started with an initial prediction of 50 points. Moreover, Table 3 highlights the number of problems students tackled during class—a significant determinant of the score. The numbers enclosed in parentheses in the table beneath Figure 2 represent actual scores, while column C2 displays scores predicted solely from C2's score (consistently 50 points). Columns 1 to 7 show predicted scores with the logs from each subsequent lesson incorporated. The average difference between the predicted and actual scores is 3.2 points. The sequence of predicted scores mirrors the order of actual scores, indicating a commendable predictive accuracy. Students A, B, E, and F are part of the training data from Section 3.3, while students B, D, and G are in the test dataset.

A comparison of the evolving predicted scores against the number of attempted problems (as depicted in Table 3) reveals that diligent effort tends to amplify the predicted score. Conversely, less effort is often accompanied by a decline. From these insights, alongside the predicted score, educators can offer targeted feedback, encouraging students to maintain their current momentum or suggesting that they tackle more problems during class. They might also advise seeking assistance from teachers or teaching assistants when required.



	C2	1	2	3	4	5	6	7
A (82)	50	57,7	70,1	68,7	81,4	85,6	82,6	84,5
B (78)	50	53	70	77,5	71,7	73,3	74,5	70,8
C (69)	50	57,8	63,5	60	72,2	78,9	74,5	69
D (58)	50	50,7	34,6	49,2	69,7	74,9	64,2	61,3
E (54)	50	47,3	49,3	47	43,8	46,8	47,9	48,8
F (41)	50	43,8	34,1	43,9	40,2	37,3	39,4	40,9
G (34)	50	37,3	38,7	37,4	36,6	37	40,2	29,7

Figure 2. Changes in the predicted scores of students with the same initial predictions

Table 3. Number of attempted problems during class

	1st class	2nd class	3rd class	4th class	5th class	6th class	7th class
A	8	6	6	5	8	7	6
B	3	6	6	6	6	6	6
C	8	7	6	5	8	7	6
D	5	0	8	7	6	7	0
E	4	5	5	5	5	6	5
F	0	0	5	4	0	6	7
G	1	5	2	5	5	6	6r

4. CONCLUSION

To utilize the programming exercise system's operation log for effective student feedback, we constructed and evaluated a multiple regression analysis model. We verified that a student's final score can be reasonably predicted from early course system logs. Furthermore, this prediction can be refreshed on a weekly basis based on the logs of each class; accordingly, pertinent feedback can be provided to each student. However, we noted an escalation in overfitting as the number of incorporated classes increased. We also deliberated on the potential for offering tailored advice aimed at enhancing the final scores of each student.

Moving forward, our intention is to consistently gather logs and reaffirm our findings through multiyear data analysis. We also plan to explore data from other courses, such as C Language Units II and IV, as well as Software Fundamentals. Our efforts will address the issue of multicollinearity and pinpoint strategies to improve the students' final scores, focusing on areas beyond in-class challenge frequency.

ACKNOWLEDGMENT

We are grateful to Professor Tsuyoshi Miyazaki, Lecturer Nozomi Nakakawaji, Lecturer Hidetoshi Okazaki, and other educators responsible for the C language class. We appreciate the invaluable feedback, bug reports, and enhancement suggestions provided by them and their students. This work was supported by JSPS KAKENHI Grant Number JP23K02747.

REFERENCES

Carter, A. et al. (2019). Leveraging the integrated development environment for learning analytics, The Cambridge Handbook of Computing Education Research. Cambridge Univ. Press, Cambridge, U.K. ch 23, pp. 679-706.

Kia, F. et al. (2020). How patterns of students dashboard use are related to their achievement and self-regulatory engagement, *Proceedings of LAK'20*, pp. 340-349.

Khulbe, M. et al. (2023). Mediating teacher professional learning with a learning analytics dashboard and training intervention, *Technology, Knowledge and Learning*, vol. 28, pp. 981-998.

Liu, E. et al. (2023). Early prediction of student performance in online programming courses. *Wang, N. et al (eds) Artificial Intelligence in Education*. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners, Doctoral Consortium and Blue Sky. AIED 2023. Communications in computer and information science, vol 1831. Springer, Cham.

Ohno, A. (2022). Evaluating the usefulness of C language learning support system as a learning analytics tool, *IIAI Information Engineering Express*, vol. 8, no. 1, pp. 1-12.

Quille, K. et al. (2019). CS1: How will they do? How can we help? A decade of research and practice, *Comput. Sci. Educ.*, vol. 29, nos. 2-3, pp. 254-282.

Raubenheimer, G. et al. (2021). Toward empirical analysis of pedagogical feedback in computer programming learning environments, *Proceedings of ACE'21*, pp. 189-195.

Satoh, A et al. (2022). Prototype of programming exercise support system able to visualize coding status, *Proceedings of 11th International Congress on Advanced Applied Informatics*, pp. 256-259.

Sedrakyan, G. et al. (2019). Guiding the choice of learning dashboard visualizations, *Linking dashboard design and data visualization concepts*, vol. 50, pp. 19-38.

Susnjak, T. et al. (2022). Learning analytics dashboard: A tool for providing actionable insights to learners, *International Journal of Educational Technology in Higher Education*, vol. 19, 12.

Sobral, S. et al. (2021). Predicting students performance in introductory programming courses: A literature review, *Proceedings of 15th International Technology, Education and Development Conference*, pp. 7402-7412.

Tanaka, T. et al. (2023a). Programming exercise system to ascertain students' coding status, *Proceedings of 11th International Conference on Information and Education Technology*, Fujisawa, Japan, pp. 204-209.

Tanaka, T. et al. (2023b). Preliminary analysis of programming exercise logs to provide effective feedback to students, *Proceedings of 12th International Congress on Advanced Applied Informatics*, Koriyama, Japan, 4 pages.