Practical Programming Education through Game Development with Unity

Mio Kobayashi*, [a], Thanyawarat Pawasopon[a], Saung Hnin Pwint Oo[a],
Yuki Yoshikawa[a], and Kamonchat Apivanichkul[a]

[a] Department of Computer Engineering, KOSEN-KMITL, Bangkok, Thailand

* mio.am@kmitl.ac.th, mio_kobayashi@shinshu-u.ac.jp

This paper discusses the implementation and evaluation of practical programming education using Unity, which is one of the most popular game development engines. Programming education is the most important engineering skill for students, regardless of their field of study in KOSEN education. The curriculum of programming education in the Department of Computer Engineering (CE) at KOSEN-KMITL is designed so that students learn Python in their 1st and 2nd years in the courses Programming 1-4. Then, in their 3rd year, they begin studying C language in Programming 5 and 6. In addition to those programming subjects, students in their 3rd year take a course called PBL, where they gain experience applying their programming skills to develop specific software in groups based on requirements provided by partner companies. Programming 7 is the final programming subject in the curriculum, offered to the 4th year students. It is a 100-minute class that runs for 15 weeks in the first semester. During the first four weeks, the class aimed to review C programming with various algorithms and was conducted as a review session of C language programming from the previous year. In the following six weeks, students learned how to develop a simple game using C# in Unity. We referred to the textbook for class material and created handouts with 43 assignment questions. By allowing students to share their progress on the assignments via a Google Sheet, students can track their progress in the class, and teachers can manage the class's overall progress. In the last five weeks, students have worked on a mini project, choosing one topic from three options: (1) customizing the simple game by adding some original functions; (2) developing a maze-solving simulator; (3) creating an original game from scratch. The number of students who chose each option was as follows: (1) 20, (2) 7, and (3) 15. In the mini project, students created a short video clip with their own narration. The evaluation was conducted using a rubric that was presented in class. Before the final submission of the video, we held a showcase presentation, where students gave feedback to each other. The results of a survey in both the first and final class, which asked 42 students whether they like programming, find it fun, and consider themselves good at it, showed that about 13% of students developed a more positive opinion by the final class.

Keywords: *Practical Programming Education, Unity, Active Learning, Game, Computer Engineering*

Introduction

KOSEN-KMITL was established in 2019 by the Thai KOSEN project, under an ODA loan agreement of the Japan International Cooperation Agency (JICA) with the Government of the Kingdom of Thailand (Aburatani et al. 2020). KOSEN-KMITL has three departments, the Mechatronics Engineering Department (ME), Computer Engineering Department (CE), and Electrical and Electronics Engineering Department (EE), and they were established in 2019, 2021, and 2023, respectively. In March 2024, the first batch of ME students graduated, and in 2025, it is the first year that the CE department has students in all grades from 1st to 5th year. The education provided at KOSEN-KMITL is equivalent to KOSEN education in Japan (Kobayashi et al. 2023). Additionally, the ME program at KOSEN-KMITL was accredited by KIS (KOSEN International Standard) in 2024.

Similar to KOSEN education in Japan, programming education is a core topic at KOSEN-KMITL, and programming skills are considered the most important engineering skill for students. In the curriculum of CE at KOSEN-KMITL, students start learning programming with Python in their 1st year in the courses Programming 1 and 2, continue their studies in 2nd year in the courses Programming 3 and 4, and learn C programming in the 3rd year in courses Programming 5 and 6. In addition to their programming courses, students in their 3rd year take a course called PBL (Project-Based Learning) (Oo et al. 2024), where they apply their knowledge and skills to developing specific software systems based on requirements provided by partner companies (Toyota Tsusho (Thailand) Co., Ltd., Neural Group (Thailand) Co., Ltd., and MapQuestAsia Corp., Ltd.). Programming 7 is the final subject offered to the 4th-year students in the programming curriculum of CE. It was a newly introduced subject of CE in 2024. Since students have already acquired basic programming skills through Programming 1-6 and the KOSEN's MCC achievement levels of the major subjects should be considered, Programming 7 needed to be designed to focus on practical applications using programming skills,

	Evaluators will be treated as anonymous, so please rate accurately.						
Session No	Choose the presenter ID	Nick Name	That presenter was explaining in their own words.(not reading written texts.) (Yes: ✓)	The presenter has a good understanding of the algorithm. (Yes: ✓)	There are no problems with the presentation attitude, voice volume, eye contact, etc. (Yes: ✓)	Choose the most excellent presenter with ✓. (Choose only ONE student.)	Comments (What you learned from the presentation.) (Required)
1	649xxxxx		✓	✓	✓		
1							

Figure 1: Report template that students submit after the presentation session.

including analysis, evaluation, and creativity. Based on this background, Programming 7 was designed using Unity, one of the most popular game development engines.

This paper discusses the implementation and evaluation of practical programming education using Unity in Programming 7. The remainder of this paper is organized as follows: Section 2 describes the structure of Programming 7 and the evaluation method. Section 3 presents the student outcomes, and Section 4 shows the survey results. Finally, Section 5 concludes the paper.

Structure of Programming 7 and Evaluation Method

Programming 7 is a compulsory subject for 4th-year students in CE. It is a 100-minute class held once a week over 15 weeks in the first semester. In 2024, forty-two students enrolled in the course. In this section, the structure of the course and the evaluation methods of students' achievements in the course are explained. Furthermore, the evaluation methods of the subject itself based on how students changed their attitude and mindset toward programming are described.

(1) Structure of Programming 7

The course is divided into three parts: Part 1: Review session of C language for 4 weeks; Part 2: Basic study of Unity for 6 weeks; Part 3: Mini project for 5 weeks.

Part 1 (4 weeks) consisted of review sessions in C language, including assignments that involved creating programs using various algorithms. The topics of each week were as follows:

- Week 1: Calculating a factorial number using recursion
- Week 2: Finding approximate solutions of polynomials by bisection and Newton Raphson methods
- Week 3: Finding optimal solutions under specific conditions using the greedy algorithm
- Week 4: Presentation for three topics

In the presentation of Week 4, three sessions of a gallery walk-style presentation were conducted. During each presentation session, audience students were assigned to make a report for each presentation based on the format shown in Fig. 1.

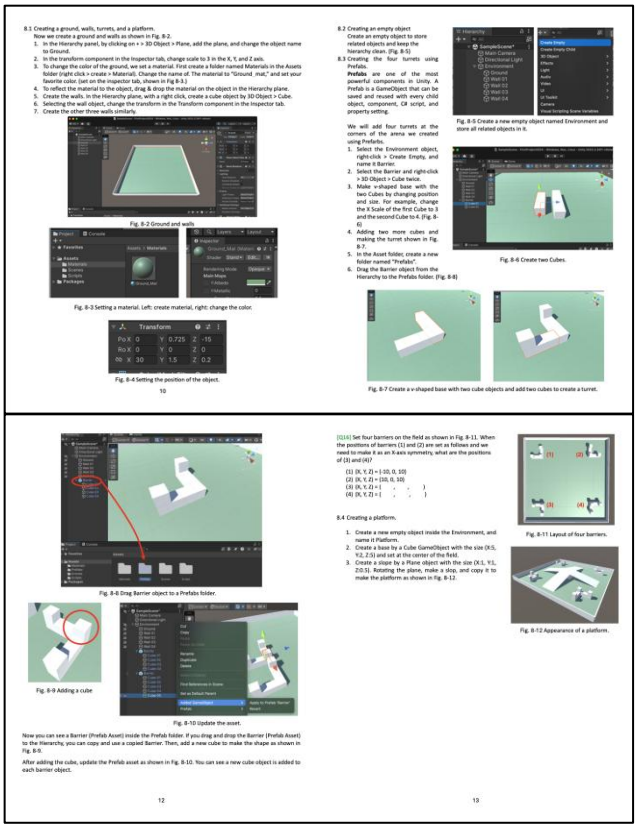


Figure 2: Part of the handout to explain how to develop a sample game.

In Part 2 (6 weeks), students learned how to develop a simple game using C# in Unity and studied the basic functions of Unity. First, they set up the environment by installing Unity on their laptop or on a desktop computer in the computer lab. The class materials were based on a textbook (Ferrone 2022), and we also created 42 pages of a handout containing 43 assignment questions. Figure 2 shows the part of the handout. The questions are grouped into 5 assignments and students submit them by each due date. Students shared their progress using a Google Sheet as shown in Fig. 3, which allowed them to track their own progress and enable teachers to monitor the overall progress of the class.

In Part 3 (5 weeks), students worked on a mini-project, choosing one topic from three options: (1) customizing the simple game by adding original features; (2) developing a maze-solving simulator; (3) creating an

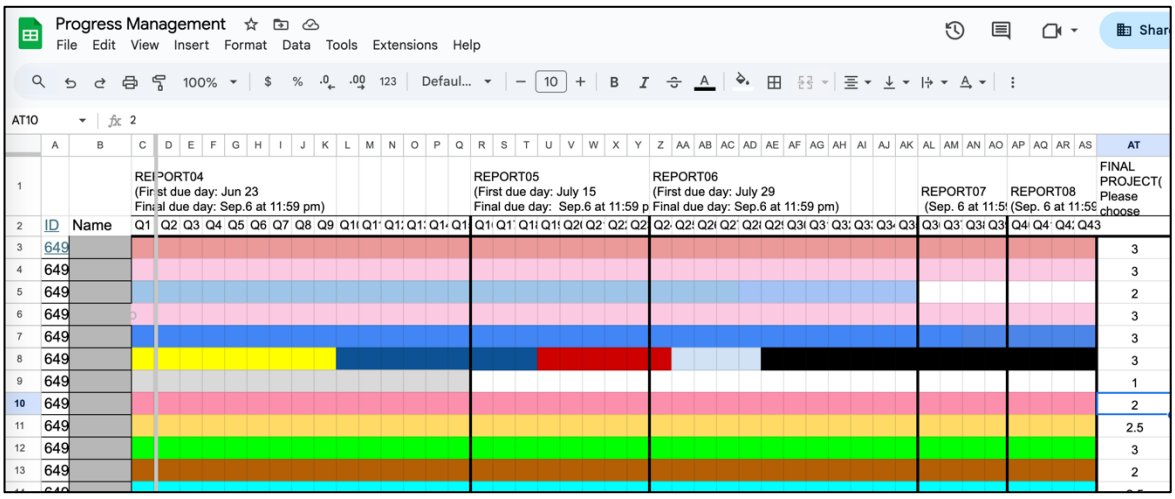


Figure 3: Progress management sheet.

	Excellent (10, 9)	Good (8, 7)	Fair (6, 5, 4)	Poor (3, 2, 1)
About requirements	Requirements for the game are specifically analyzed and designed in detail. The development are excellently achieved.	Requirements for the game are analyzed and designed. The development are achieved.	Requirements for the game are analyzed. The development are achieved.	Requirements for the game are not analyzed. The development are achieved only some parts.
Report	The format is appropriate. Including concept, gameplay, interface, and art style. The description is clear and logical. The game idea is unique and nice.	The format is appropriate. Including concept, gameplay, interface, and art style. The description is clear and logical.	The format is appropriate. Including concept, gameplay, interface, and art style.	The format is not appropriate. Not Including concept, gameplay, interface, or art style.
Presentation (Short video)	Presentation is logical and appropriately explained. Duration is less than 1 min. and not too short.	Presentation is logical and appropriately explained. Duration is less than 1 min.	Presentation is good.	Delay due day.

Figure 4: The rubric for the mini project: development of an original game from scratch.

original game from scratch. We provided these three options because students' progress varied depending on their programming skills and their familiarity with tools like Unity. Although we allocated 6 weeks for developing a simple game (Part 2) and 5 weeks for a mini project (Part 3), it was supposed that some students would be unable to complete Part 2 within the given time, while others finished it in less than 5 weeks. Students who needed more time to complete Part 2 were encouraged to choose option (1), which required less time compared to options (2) and (3). On the other hand, students who completed Part 2 quickly could move on to Part 3 early and were encouraged to choose option (2) or (3), which allowed for more detailed and creative development. Regarding option (2), students had previously learned various maze-solving algorithms in a subject titled Introduction to Artificial Intelligence. The purpose of this option was to allow students to implement those algorithms in the form of a game application.

In the 5th week of Part 3, a showcase session was held to review all games currently in development and to provide mutual feedback. In the class, three sessions were held, and students were divided into nine groups for each session. The group members were arranged in such a way that no group had the same members across different sessions. For the mini-project, students were required to submit a Final Report including a report of the mini-project, a one-minute video of their game, and the exported project file from the Unity engine.

The submission deadline for the mini project was set for two weeks after the showcase session, giving students enough time to revise or continue their development based on the feedback received from classmates and teachers during the session.

(2) Evaluation methods of students' achievements

The evaluation of students' achievements in Programming 7 is based on five criteria: an examination (20%), quizzes (10%), mutual evaluations between students (20%), reports (30%), and other components (20%).

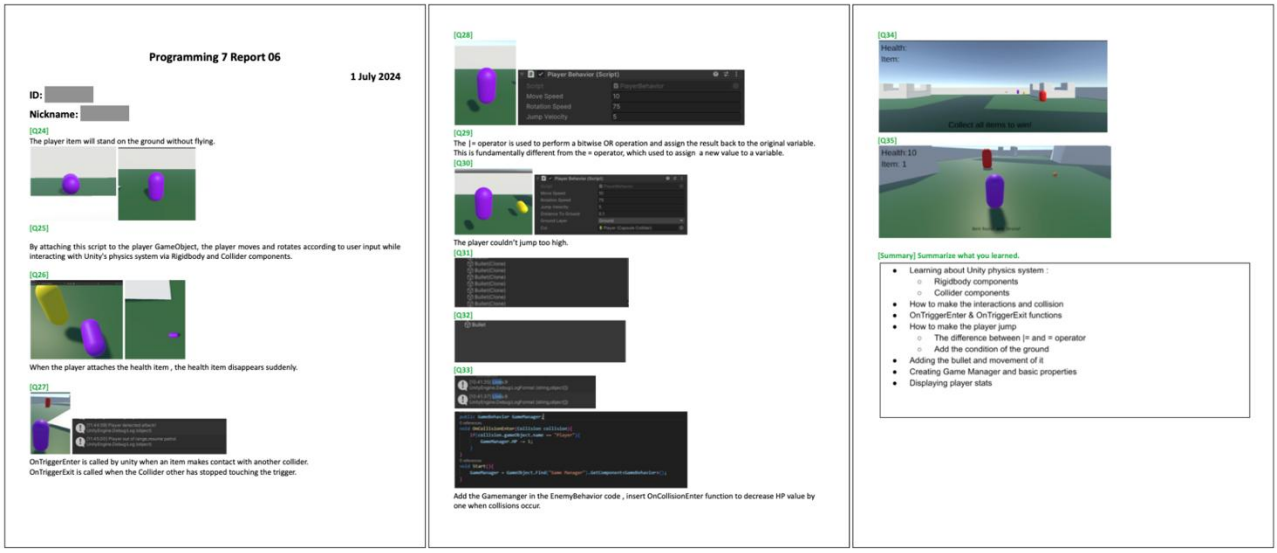


Figure 5: Examples of reports submitted by a student.

Regarding the examination, only a midterm exam was conducted, which tested students on the basic review topics of C programming covered in class, as well as fundamental knowledge of Unity. As for the quizzes, only one was administered at the end of the course during this term.

Mutual evaluation is related to the presentation session in Part 1 and the showcase session in Part 3. During the presentation and showcase sessions, students voted for the most valuable presenter in each session or group, the results were used for mutual evaluation.

In this course, students were assigned a total of 10 reports, consisting of three reports in Part 1, one activity report in the presentation session in Part 1, five reports in Part 2, and the Final Report in Part 3. Except for the Final Report, reports were evaluated based on whether they were submitted because the contents are explained in the class and the reports are just the evidence that students worked on the development of each step. However, if a report is deemed unacceptable, for example, it is almost blank, uses an inappropriate format, or includes some problems, students must revise and resubmit it. If a report was not perfect but it was acceptable, it was accepted with a few point deductions. The Final Report, consisting of a report of the mini-project, a one-minute video of their game, and the exported project file from the Unity engine, was evaluated by the rubric presented in the first class of Part 3. Since three options for the mini project were offered, three rubrics were prepared. Figure 4 presents the rubric for the option (3): creating an original game from scratch.

As for other components, we considered students' behavior based on whether students submitted assignments on time or late. A 10-point deduction was applied when students submitted each assignment late.

(3) Evaluation methods of how students changed their attitude and mindset toward programming

To evaluate how students' attitudes and mindsets toward programming changed after taking Programming 7, we conducted surveys at both the beginning and end of the course. In the first survey, students were asked four questions: "Do you like programming?", "Do you think programming is fun?", "Are you good at programming?", "What do you think is the reason for learning programming?" The first three are closed-ended questions with three response options: "Yes.", "No.", and "I am not sure." The fourth is an open-ended question that requires a free-writing response. In the last survey, in addition to the same four questions from the first survey, an open-ended question, "Do you like Unity programming?", was added, which required a free-writing response.

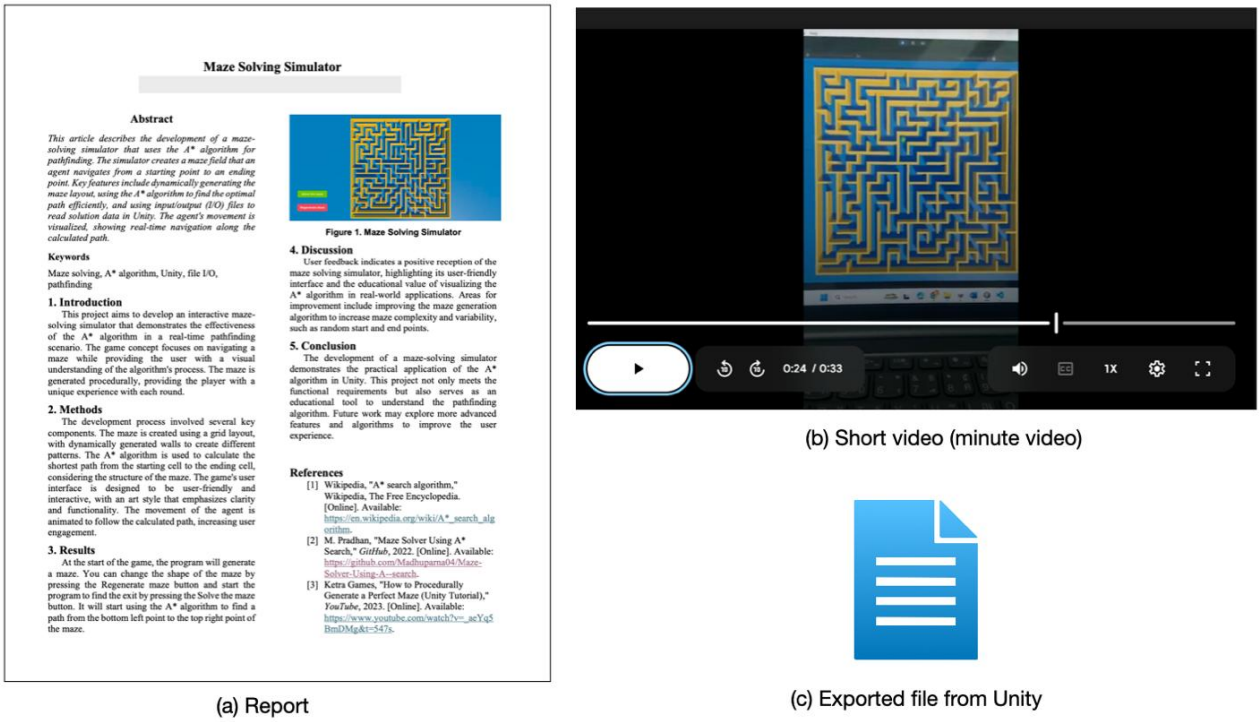
To analyze the results, we focused on the three closed-ended questions and created a state transition diagram with four states: "Yes," "No," "Not sure," and "N/A." The results are presented and discussed in the Survey Results section below.

Student Outcomes

Figure 5 shows an example of reports submitted by a student about developing a simple game using Unity (Part 2). As for the report template, blank formats are prepared, and students can use the template and make a report while creating a sample game by following the handouts. As mentioned in the previous section, the student's progress in the creation is monitored by the progress management sheet shown in Fig. 3, and some students whose progress is delayed can be identified and guided by teachers.

Figure 6 shows the three items, a report, a minute video, and a exported files from Unity, that students had to submit as the outcomes of the mini-project (Part 3). Figure 7 shows the situation during the showcase session held in Part 3. Students gave a feedback each other.

In the mini project, the number of students who chose each topic from three options was (1) 20, (2) 7, and (3) 15. About topic (1), students successfully added multiple functions to the original game studied in Part 2, for example, changing the shape of the game field and adding multiple characters. Regarding topic (2), only 7 students worked on the topic. However, the quality of their work was very high, and all students could



(a) Report

(b) Short video (minute video)

(c) Exported file from Unity

Figure 6: Assignment for the mini-project: (a) report, (b) short video of a mini-project, and (c) exported file from Unity.

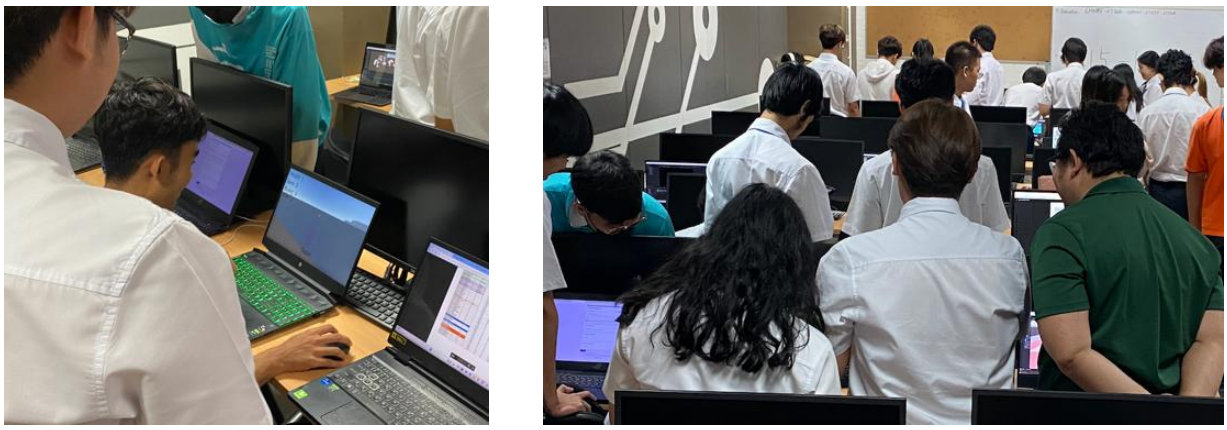


Figure 7: Situation during the showcase session.

implement A* algorithm that is one of the maze-solving algorithms they have learned in other class. About topic (3), some students created a game which has several advanced functions that were not taught in the previous classes of Part 2. Among those students, some students tried to create 3D models with Blender to use them in their developing game and used a head-mounted display in their presentation. It can be said that these students actively learned more on their own than the knowledge provided in the class.

Survey Results

Table 1 summarized the results of surveys that were conducted at both the beginning of and end of the course, focusing on the three closed-ended questions. In the first survey, all 42 students answered, however, 38 students answered at the end survey. To analyze the results, we use a state transition diagram with four states: “Yes,”

Table 1: Summary of the survey results. 42 and 38 students answered to the first and last surveys, respectively.

Q1: Do you like programming?

	First Survey	Last Survery
Yes.	28	28
No.	3	2
I am not sure.	11	8

Q2: Do you think programming is fun?

	First Survey	Last Survery
Yes.	25	27
No.	4	2
I am not sure.	13	9

Q3: Are you good at programming?

	First Survey	Last Survery
Yes.	5	8
No.	11	10
I am not sure.	26	20

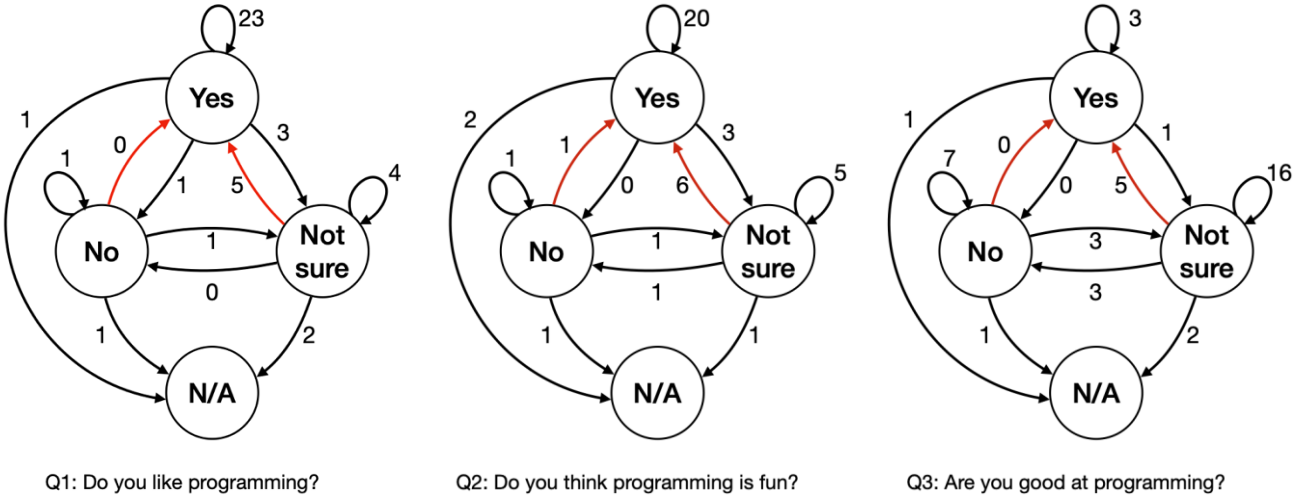


Figure 8: State transition diagrams based on students' responses to three closed-ended questions. The number labeled on each arrow indicates the number of students.

"No," "Not sure," and "N/A." Figure 8 shows a state transition diagrams about each question. In Fig. 8, the number labeled on each arrow indicates the number of students. Although there are 12 types of transitions, such as from "No" to "Yes" and from "Not sure" to "No", we consider only from "No" to "Yes" and "Not sure" to "Yes" are positive changes in students' attitude toward programming, as indicated by the red arrows in Fig. 8. From the analysis, we found that an average of 13.4%, calculated from 11.9% (Q1), 16.7% (Q2), and 11.9% (Q3), of the 42 students developed a more positive opinion by the final class. Especially, about the question, "Q2: Do you think programming is fun?", the ratio of positive change was highest. It can be concluded that students who found creating a game enjoyable and had fun developing their game with Unity though Programming 7 showed a positive change in their attitude and mindset toward programming.

Conclusions

This paper discussed programming education conducted in the Programming 7 for the fourth-year students in the Department of Computer Engineering at KOSEN-KMITL. Unity, one of the most popular game development engines, was used as the programming platform. In the Programming 7, after taking classes of reviewing C language and learning basic usage of Unity, students worked on the mini project, creating a game based on a theme they selected from three options. Students' achievement in the mini project was evaluated using a rubric that was presented in the class. Most students worked proactively on the project and presented their games during the showcase session. To evaluate how students' attitudes and mindsets toward programming changed after taking Programming 7, we conducted surveys at both the beginning and end of the course and analyzed the results of three closed-ended questions, focusing on the change in students' responses. Based on the analysis, we found that an average of 13.4% of the 42 students developed a more positive opinion toward programming by the final class of Programming 7. We concluded the reason of this results is that students who experienced enjoyment in creating games developed a more positive perception of programming.

This programming course was implemented for only one academic year so far. Therefore, the survey results might not be enough to discuss the effectiveness of the course. To allow for more accurate discussion, survey results over multiple years should be collected and analyzed. In this study, although we asked an open-ended question, "What do you think is the reason for learning programming?", only three closed-ended questions were analyzed. Investigating the responses to the open-ended question remains a task for the future.

References

Aburatani, H., Sittichivapak, S., Kano, S. and Uehara., N. (2020). Curriculum and Implementation of KOSEN Eengineering Education at KOSEN-KMITL, Thailand, Proceedings of the 16th International CDIO Conference, hosted on-line by Chalmers University of Technology, pp. 54-63, Gothenburg, Sweden, 8-10 June 2020.

Kobayashi, H., Pawasopona, T., Tangsuknirundorna, P., Keatsamarna, T., Rittiplanga, A., Srithanasarna, K., Oo, S.H.P., Paraa, M., Yoshikawa, Y. and Doi, S. (2023). Modifying a "Reverse Engineering" Class for the Department of Computer Engineering at KOSEN-KMITL. Proceedings of 16th International Symposium on Advances in Technology Education (ISATE) (pp. 324-327), September 12-15, 2023, Matsue City, Japan.

Oo, S.H.P, Pawasopon, T., Tangsuknirundorn , P., Keatsamarn, T., Srithanasarn, K., Su Wai Myo, S.W., Shah, S., Kobayashi, H., Kobayashi, M. and Yoshikawa, Y. (2024). Establishing Automated Attendance Checking Systems through Project-Based Learning in KOSEN-KMITL. Proceedings of 17th International Symposium on Advances in Technology Education (ISATE) (pp. ST3-136-ST3-141), September 24-27, 2024, Singapore.

Ferrone, H. (2022). Learning C# by Developing Games with Unity: Get to grips with coding in C# and build simple 3D games in Unity 2023 from the ground up 7th ed. Edition, Packt Publishing.