International Journal of Learning, Teaching and Educational Research
Vol. 23, No. 7, pp. 347-367, July 2024
https://doi.org/10.26803/ijlter.23.7.18
Received May 25, 2024; Revised Jul 15, 2024; Accepted Jul 18, 2024

Diving into the Future: Unravelling the Impact of Flowgorithm and Discord Fusion on Algorithm and Programming Courses and Fostering Computational Thinking

Rizki Hardian Sakti, Nizwardi Jalinus, Sukardi, Hendra Hidayat
Rizky Ema Wulansari*
Universitas Negeri Padang
Padang, Indonesia

Chau Trung Tin
Nazarbayev University
Astana, Kazakhstan

Firas Tayseer Mohammad Ayasrah
College of Education, Humanities and Sciences, Al Ain University
United Arab Emirates

Abstract. Algorithm and programming have a close relationship, and this type of course can develop the computational thinking skills needed by students in the current digital era. Meanwhile, there are some challenges to teaching algorithm and programming, including engaging the students' motivation and interest. Thus, Flowgorithm and Discord have been chosen to be implemented in this course. This is because these two tools can help beginners understand algorithms and programming easily. This study aims to examine the influence of integrating Flowgorithm and Discord as part of an Algorithm and Programming course, with a focus on enhancing computational thinking skills. This research employed a quasi-experimental method, with a total sample of 32 students for the experimental group and 35 students for the control group, who were randomly selected using the simple random sampling technique. The data was collected through knowledge assessments, classroom observations, and unstructured student interviews, and the answers were analyzed using the t-test and multiple linear correlation. The findings demonstrate that the utilization of Flowgorithm and Discord in conjunction yielded positive outcomes in terms of enhancing the student's computational thinking skills. The combination of these two

*Corresponding author: *Rizky Ema Wulansari, rizkyema@ft.unp.ac.id*

tools supports more interactive, collaborative and effective learning, and enhances the development of computational thinking skills that are essential in today's world of education and the technology industry.

It can be concluded that the combined use of these tools can have a significant impact when it comes to improving the effectiveness of algorithms and programming. Therefore, this study only focuses on enhancing the computational thinking skill. Further research needs to be conducted to test its applicability across different thinking skill and learning contexts.

Keywords: Flowgorithm; Discord; Algorithm and Programming; Computational Thinking Skill

1. Introduction

Education in algorithms and programming has an increasingly important role in preparing future generations to face the challenges of this digital age. As technology advances, an understanding of algorithms, programming, and computational thinking are becoming highly valued skills in a variety of fields, including science, technology, engineering, and business (Deveci Topal et al., 2017). Amidst the rapid development of information and communication technology, an understanding of algorithms, programming and computational thinking is becoming increasingly important when it comes to preparing future generations for the challenges of this digital era. Schools and colleges around the world are increasingly recognizing the need to provide adequate education in these areas to enable students to compete in an increasingly competitive and rapidly changing job market (Chen et al., 2013).

However, teaching algorithms and programming is often faced with challenges, especially when it comes to motivating and engaging students who may be newly exposed to the concepts. Therefore, innovative and up-to-date approaches to learning are needed to improve the effectiveness of teaching and learning in this area (Graham et al., 2020). Teaching algorithms and programming is not always easy for students, especially for those who are newly exposed to the concepts. To improve the understanding and interest of students in this field, innovative approaches to learning are needed. One promising approach is the use of engaging and interactive software or platforms (Lee, 2013). Based on observations and interviews, students have implied that they need a platform that can motivate them to learn algorithms and programming.

Due to this, there are two tools that stand out in the context of teaching algorithms and programming, namely Flowgorithm and Discord. Flowgorithm is a software tool used to create and test algorithms using flow charts, which makes it easy for beginners to understand the concept of algorithms. Discord is a popular online communication platform, which provides features such as text, voice, and video chat, as well as the ability to share screens. Previous research has explored various approaches to teaching algorithms and programming, including the use of a range of software, online platforms, and learning methods. However, there is still room for improvement in the effectiveness of teaching and learning in this area (Bernacki et al., 2020).

Previous research has used conventional approaches in the teaching of algorithms and programming, such as lectures and written exercises (Lai & Jin, 2021). This research, on the other hand, proposes a more innovative approach by integrating tools such as Flowgorithm and Discord. The integration of these advanced technologies is expected to increase student engagement and learning effectiveness (Christensen & Knezek, 2016). Some of the previous studies have also explored the use of visualization tools to aid the understanding of algorithmic concepts (Florea et al., 2010). However, the use of Flowgorithm in this study provides a stronger visualization dimension by using easy-to-understand flowcharts. This can assist students in understanding the algorithm steps visually and intuitively.

This research therefore aims to investigate the impact of combining Flowgorithm and Discord in teaching algorithms and programming, as well as improving their overall computing skills. Because Algorithms and Programming courses have a close relationship with computational thinking skills, they complement each other in developing the analytical and technical skills needed to solve problems efficiently and effectively (Lai et al., 2021). This course develops computational thinking skills including problem analysis, solution design, and solution implementation, as well as the evaluation and optimization of solution performance. Thus, Algorithms and Programming not only provide technical skills but also form a logical, analytical and systematic way of thinking which is the essence of computational thinking (Romero et al., 2017).

In addition, this research can also serve as a basis for the development of more innovative and interactive teaching approaches in the field of computing. One of the novelties of this research is its emphasis on using online communication platforms such as Discord to facilitate collaboration and interaction between students and teachers (Gupta, 2023). This approach creates a more dynamic and interactive learning environment, which has been lacking in the previous research. This research also actively utilizes technology, both in terms of the use of learning software (Flowgorithm) and online platforms (Discord). This reflects a response to the rapid development of information and communication technology, which enables a more dynamic and up-to-date approach to learning.

2. Literature Review

Algorithms and Programming courses have a close relationship with computational thinking skills, because they complement each other in developing the analytical and technical skills needed to solve problems efficiently and effectively (Ezeamuzie et al., 2022). In the Algorithms course, students learn to break down complex problems into smaller, easier-to-solve parts, identify recurring patterns, ignore irrelevant details, and design detailed steps to solve the problem. Meanwhile, Programming courses involve translating algorithms into code that can be executed by a computer, identifying and correcting errors (debugging), testing code to ensure that the solution runs correctly, as well as continuously iterating and refining solutions for better efficiency (Nouri et al., 2020). This course develops computational thinking skills including problem analysis, solution design, and solution implementation, as well as the evaluation and optimization of solution performance (Lemay et al., 2021). Thus, Algorithms and Programming not only provide technical skills, but the course also forms a logical, analytical and systematic way of thinking that is the essence of computational thinking.

Computational thinking skills is a concept that involves problem-solving approaches inspired by the thinking and processes used in computing. It involves using computational principles to solve problems systematically and logically, even if the problem is not directly related to computers or programming (Gao & Hew, 2022). Computational thinking skills are not limited to the domain of computers or programming. They can be applied in a variety of contexts, including science, math, design, economics, and more. The development of computational thinking skills assists individuals in developing logical, analytical and creative thinking, which is essential in facing challenges and solving problems in an increasingly technologically connected world (Gong et al., 2020).

In order to develop computational thinking skills, education and higher education institutions play an important role. Curricula can include learning computing and programming, as well as an emphasis on problem solving, data analysis and modeling (Zha et al., 2020). In addition, the use of computational technologies and tools in the learning process can also facilitate the development of computational thinking skills.

Computational thinking skills are becoming increasingly important in this digital age. Preparing individuals with these skills enables them to become innovators, effective problem solvers and active players in an ever-evolving world. The implementation of computational thinking skills in higher education helps produce graduates who are ready to face real-world challenges (Cansu & Cansu, 2019). A curriculum integrated with computational thinking skills prepares individuals to be critical, innovative, and adaptive problem solvers in an era dominated by technology and complexity (Li et al., 2020). The application of computational thinking skills also has wide-ranging benefits, including analytical abilities where computational thinking skills involve the ability to analyze situations, identify patterns, and evaluate solutions. This assists individuals in developing strong analytical thinking, which is useful when facing complex challenges and making informed decisions (Liao et al., 2022; Nouri et al., 2020).

Implementing Flowgorithm and Discord can help develop computational thinking skills, such as problem decomposition, pattern recognition, abstraction, and algorithm design. With Flowgorithm, you can visualize the decomposition process in the form of a flowchart and depict important aspects of the algorithm without getting bogged down in detailed implementation code. Meanwhile, Discord enables live discussion and collaboration, where participants can work together on flowcharts, share screens for immediate reflection, and get guidance and input from mentors or peers. Combining the visualization power of

Flowgorithm and the collaboration capabilities of Discord creates a learning environment that supports the effective development of computational thinking skills.

3. Methods

3.1 Research Type and Design

This research is quantitative and descriptive. This research design uses a quasi-experimental approach, using pre-test and post-test experiments, as well as a control-experiment group design for analyzing the effectiveness of the coding task and user experience. The pre-test and post-test are perfectly designed to demonstrate the effectiveness of Flowgorithm and Discord, when one group is randomly assigned, and the other group is given a treatment that has been prepared (Christensen, 2001). Two homogeneous groups were selected to make up the sample, consisting of an experimental group and a control group. The design is shown in Figure 1. The survey method was used to collect the correlations of computational thinking data.

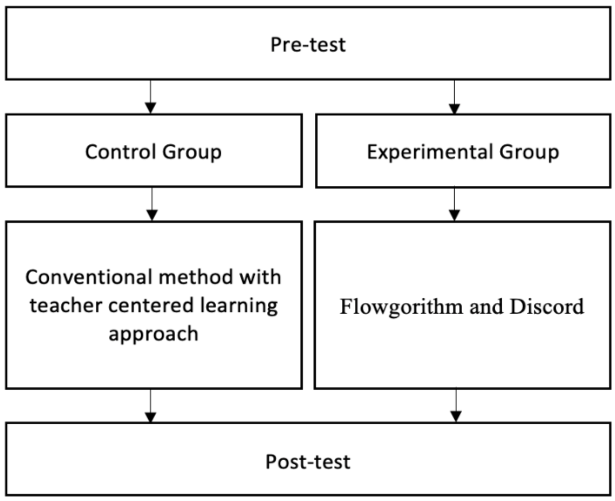


Figure 1: Research design

3.2 Research Procedures

The study began with the selection of a sample of students from one or more classes who were then randomly divided into two groups: a control class and an experimental class. Before treatment, a pre-test was given to all students to measure their initial knowledge of algorithms and programming. For the experimental class, the treatment started with the first session introducing Flowgorithm to students, followed by basic exercises to understand the interface and basic functionality. In the second session, the students implemented a simple algorithm using Flowgorithm, where discussions and debriefs were conducted via Discord, allowing students to collaborate and share ideas. The third session involved working on small projects with more complex algorithms using Flowgorithm, with communication and collaboration continuing through Discord.

Meanwhile, the control class received lessons on algorithms and programming using traditional methods without the use of Flowgorithm and Discord. In the first session, teaching was done through lectures and written exercises. The second session continued teaching simple algorithms using face-to-face discussions. In the third session, the students in the control class worked on small projects using the same traditional methods, with collaboration and discussion conducted directly in class. After the intervention, a post-test was given to all students to measure the improvement in their knowledge of algorithms and programming. In addition, qualitative data was collected through observations during the learning sessions in both groups to record the interactions, engagement, and classroom dynamics, alongside interviews with several students and teachers to gain in-depth insights into their experiences during the study. Perception questionnaires were also distributed to students from both groups to gauge their perceptions of the applied learning methods.

3.3 Research Sample

The population of this study was all PTI study program students enrolled in the 2023/2024 academic year. The sample selection technique used was the simple random sampling technique by selecting a randomization sample, which can be achieved at two levels (Lohr, 2010). The randomization of the sample is achieved through the selection of subjects from a homogeneous population. In this study, the first randomization requirement was selected in two stages of cluster sampling technique (Lohr, 2010); The first stage was the selection of universities. The population of this research was universities that have Informatics and Computer Engineering Departments in West Sumatra. In the first stage, one university was randomly selected, namely Universitas Negeri Padang. In the second stage, two classes were randomly selected from the university as the experimental group and control group. Then, one class was chosen for the experimental group and one class for the control group at Padang State University. According to Christensen (2001), as each subject has the same opportunity to be selected to be a sample from the entire population, the assignment of control groups and experimental groups is not based on certain characteristics. Thus, two classes attending Algorithm and Programming courses among PTI study program students were selected, with the total being 32 students for the experimental group and 35 students for the control group. This research emphasizes integrity, transparency and ethics in research involving humans, where it will be ensured that the rights and welfare of research subjects are strictly protected. This research obtained approval from the research subjects to be used as the research sample.

3.4 Research Instruments

The research instruments used included an achievement test, problem solving and computational thinking questionnaires (pre-test and post-test) to determine effectiveness and collect the user experience data. The instrument used to measure computational thinking skill was the computational thinking skill assessment rubric developed by George Mason University. This rubric was used because the rubric is intended for students in the field of engineering (Peters-Burton et al., 2018), using a 1-4 Likert scale. The instrument grid for the computational thinking skill assessment rubric can be seen in Table 1. The problem-solving skill questionnaire consists of solution, planning, preparation, processing and results indicators, while the achievement test consists of array, loop, sorting and matrix indicators.

Table 1: Computational thinking skill questionnaire

No.	Indicator	Description
1	Pattern abstraction and generalization	Examine a group of patterns and describe them in a clear and efficient way
2	Systematic information processing	Using heuristics to understand an event
3	Symbol and representation system	Describing events that are often abstract with simplified concretes
4	Algorithmic notion of control flow	Manage data using certain procedures
5	Structured problem decomposition	Break complex problems or systems into more understandable parts
6	Iterative, recursive and parallel thinking	Repeating thoughts in cycles to achieve a goal (iterative), thinking through thoughts (recursive), and the ability to focus thoughts in a specific direction (parallel)
7	Conditional logic	States that the occurrence of one event depends on the occurrence of another event
8	Efficiency and performance constraints	Considering the hindering and favorable factors involved in a process
9	Systematic debugging and error detection	Engage in a methodical process to find and reduce defects

3.5 Research Validity Control

3.5.1 *Internal validity control*

To ensure internal validity in this study, various strategies were implemented. First, the students were randomly divided into a control group and a treatment group to reduce selection bias. Second, the two groups were matched based on demographic and academic characteristics such as age, gender, and level of prior knowledge of algorithms and programming, ensuring group equality. External variables such as the learning environment, teaching materials, and the time given to both groups were controlled so then only the teaching methods differed. The use of standardized test instruments and questionnaires with proven validity and reliability ensured measurement accuracy. In addition, blinding was done so then the teacher or assessor did not know which group was given the treatment to reduce assessment bias. Measurement of the students' knowledge was conducted before (pre-test) and after (post-test) the intervention to see whether the changes that occurred did so due to the intervention. Observations during the learning sessions were conducted to ensure that the intervention was implemented as planned and to identify other factors that may have affected the results.

3.5.2 *External validity control*

For external validity, this study selected a representative sample of the target population by ensuring diversity in terms of the students' academic ability, technological background, and demographic characteristics. A detailed description of the research context, including sample characteristics, learning settings, and methods used, has been provided to help other researchers understand the research conditions and apply the results in a similar context. This study also recommends replication in different settings and with different samples to test the generalizability of the findings. The use of tools and methods that have been recognized in the field of computer education will increase the likelihood that the research results can be compared with previous studies and applied in various contexts. The results of this study will be disseminated through scientific publications and presentations at conferences, with discussions on how these results can be applied in various educational settings and how different contexts may affect the effectiveness of the intervention.

3.6 Pilot Study

The pilot study aimed to test the feasibility and reliability of the research instruments as well as to identify potential technical and logistical problems before conducting the main study. In addition, the pilot also aimed to collect preliminary data that would be used for an initial evaluation of the effectiveness of combining Flowgorithm and Discord in learning algorithms and programming. The results of the pilot study were analyzed using the Cronbach's alpha coefficient formula, where the computational thinking skills questionnaire had a reliability value of 0.715, which means that the questionnaire was determined to be reliable and could be used for the data collection activities.

3.7 Data Analysis Technique and Hypothesis Development

The data analysis in this study included quantitative data collection and analysis to provide a comprehensive understanding of the impact of combining Flowgorithm and Discord in the learning of algorithms and programming. Quantitative data was collected through tests (pre-test and post-test) that measured the students' computational thinking skills before and after the intervention. Descriptive analysis was conducted to calculate the mean, median, and standard deviation of the pre-test and post-test scores, while the inferential analysis used t-test and multiple linear correlation to compare the mean scores between the control and treatment groups. This aims to determine whether there was a significant difference in the students' computational thinking skills due to the use of Flowgorithm and Discord. Broadly speaking, the hypotheses in this study are as follows:

H01: There is no statistically significant difference in the mean scores of the students' computational thinking skills between the control group and the experimental group.

H02: There is no statistically significant relationship between Discord and Flowgorithm and the students' computational thinking skill.

4. Results

4.1 Error Rate in the Algorithm and Programming Tasks

Error rate was calculated as the percentage of errors out of the total instructions or steps in the algorithm implementation. The errors were grouped into categories such as syntax errors, logic errors, and runtime errors, and the error rate comparisons between students in the experimental class and control class were analyzed using statistical methods such as the t-test to determine the significance of the differences. By measuring the error rates of the LCS programming tasks, empirical evidence was provided of the positive impact of using Flowgorithm and Discord in the context of algorithm and programming education. The finding that students in the experimental class experienced lower error rates would suggest that this innovative teaching approach is more effective than traditional methods. Flowgorithm helps students visualize algorithm logic more clearly, while Discord provides a platform for collaboration and discussion that can accelerate the process of problem solving and error detection. In addition, the analysis of the most frequent error types is able to provide educators with additional insights into the specific areas where students need the most support. For example, if logic errors are found to be most dominant, educators can focus on reinforcing programming logic concepts through additional exercises or using Flowgorithm to visualize the logic flow in more detail. If syntax errors are more dominant, it may be necessary to increase the coding practice with immediate feedback via Discord.

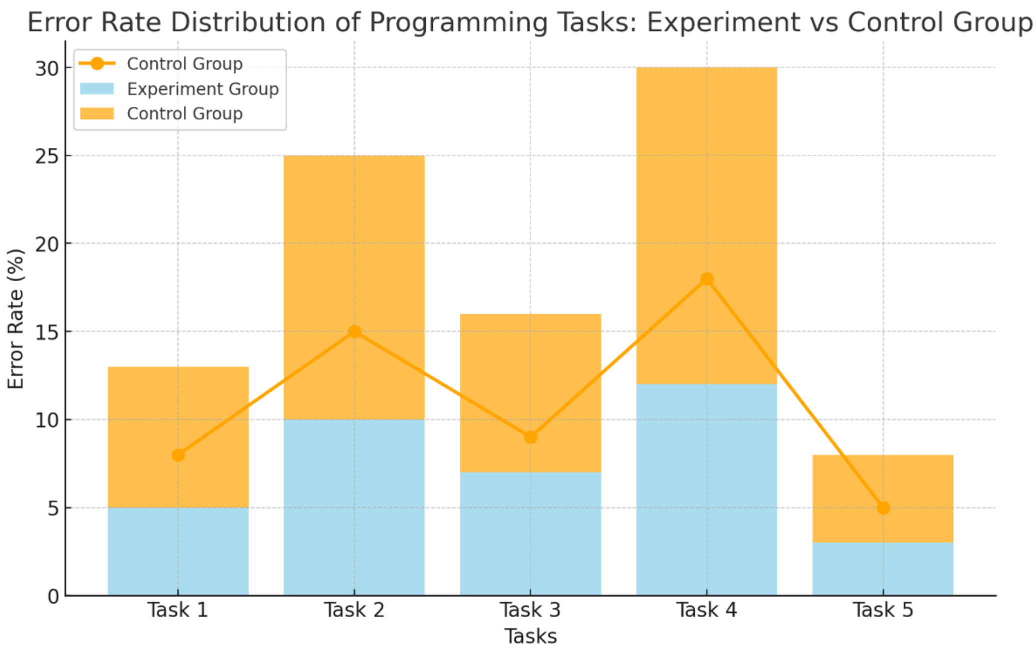


Figure 1: Error rate distribution of the programming tasks

The results of this study showed there to be a significant difference in error rate between the students in the experimental and control classes for the programming task using the Longest Common Subsequence (LCS) algorithm. The results showed that students in the experimental class had a lower error rate compared to students in the control class, which would indicate that the use of Flowgorithm for algorithm visualization and Discord for collaboration helped to reduce programming errors. Students in the experimental class, who used Flowgorithm to visualize algorithms and Discord for collaboration, showed lower error rates compared to students in the control class who used traditional learning methods.

In the experimental group, the most common types of errors were logic errors in algorithm implementation, while the number of syntax and runtime errors were lower compared to the control class. For the control class, the most common types of error were logic errors and syntax errors, and runtime errors occurred more frequently due to the lack of in-depth understanding of the logic of the algorithm assisted by Flowgorithm visualization in the experimental class. The results of this study also show that the students in the experimental class felt more helped by the step-by-step visualization provided by Flowgorithm. They could see how the LCS algorithm worked dynamically, which helped them understand the logic behind the code they had written. Discussions through Discord allowed them to immediately ask questions and get answers to the difficulties they faced, thus reducing the mistakes they made.

4.2 Effectiveness of the Coding Task in the Treated Group

The effectiveness assessment of the coding task was divided into two assessments, namely the algorithm design performance assessment using Flowgorithm software and the coding performance assessment using C++. Based on Table 2, the experimental group students in the performance assessment of algorithm design and coding tasks using Flowgorithm and Discord software had a performance assessment presentation score above and equal to 75. This means that Flowgorithm and Discord are effective at helping students design algorithms.

Table 2: Performance assessment data for the algorithm design and coding task in the treated group

		Solution planning	Preparation	Processing	Results
Case 1	Group 1	83,33	87,5	93,75	85
	Group 2	91,67	87,5	81,25	85
	Group 3	83,33	87,5	93,75	95
	Group 4	91,67	100	87,5	90
	Group 5	83,33	87,5	93,75	95
Case 2	Group 1	75	87,5	81,25	80
	Group 2	91,67	75	87,5	85
	Group 3	75	87,5	81,25	90
	Group 4	75	100	75	95
	Group 5	83,33	75	100	85
Case 3	Group 1	91,67	75	81,25	85
	Group 2	83,33	87,5	87,5	85
	Group 3	83,33	100	81,25	80
	Group 4	83,33	75	87,5	90
	Group 5	91,67	100	75	95
Case 4	Group 1	83,33	75	81,25	85
	Group 2	83,33	100	81,25	85
	Group 3	75	87,5	93,75	90
	Group 4	83,33	100	81,25	90
	Group 5	75	87,5	75	95

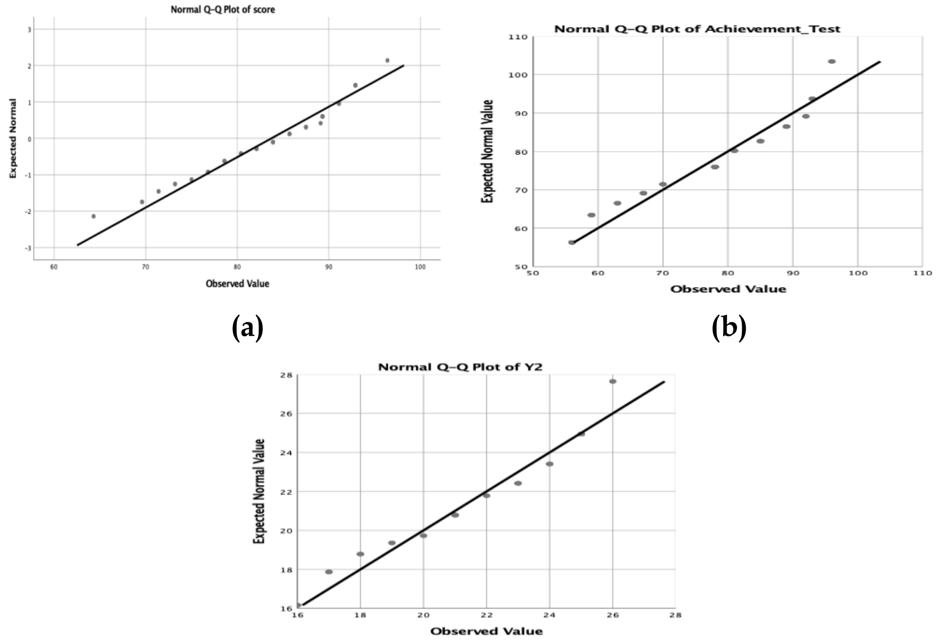
The effectiveness of the programming tasks in the treatment group using Flowgorithm and Discord was evaluated in depth. The results show that students in the treatment group had a lower error rate for the programming task compared to the control group using traditional learning methods. The average error rate in the treatment group was 15%, while in the control group, it was 35% based on the LCS measurement, suggesting that the visualization and collaboration provided by Flowgorithm and Discord helped the students understand and apply algorithms more precisely. In addition, the students in the treatment group showed a better understanding of the algorithm concepts, with Flowgorithm making it easier to visualize the algorithm steps, reducing confusion and improving accuracy in the code's implementation. The quality of the code produced by the students in the treatment group was also cleaner and more structured, as Flowgorithm allowed them to design algorithms more systematically before implementing them. The Discord platform, on the other hand, provided a space for the students to discuss and help each other in completing the programming tasks, accelerating the problem-solving and learning process through screen sharing and immediate feedback.

The students' motivation and engagement with the programming tasks also increased with the use of Flowgorithm and Discord. The visualization of algorithms makes learning more interesting, while the social interaction over Discord makes the learning process more fun and supportive, so the students are more motivated to complete tasks and achieve better results. Overall, this study shows that the combination of Flowgorithm and Discord has a significant positive impact on learning algorithms and programming, improving the students' programming competencies and developing deeper computational thinking. The findings make a valuable contribution to information and communication technology education, demonstrating that this innovative approach can effectively support and improve the learning process.

It is also shown that the use of these technologies can facilitate the development of better problem-solving skills. With Flowgorithm, students can break down complex problems into smaller and more manageable steps, while with Discord, they can discuss different strategies and approaches with their peers. This combination helps students to think more critically and systematically when dealing with programming tasks, which is one of the key competencies in computational thinking. Overall, this study provides strong evidence that the integration of Flowgorithm and Discord in learning algorithms and programming can improve the effectiveness of programming tasks in the treatment group. These findings emphasize the importance of using technologies that support visualization and collaboration in computing education. By adopting these technologies, educators can create a more dynamic and interactive learning environment, which not only improves the students' concept understanding and programming skills but also builds on the collaborative skills and computational thinking that are essential for success in today's digital world. This research paves the way for further exploration into the use of innovative technologies in education and how they can be effectively integrated to improve student learning outcomes.

4.3 User Experiences of Flowgorithm and Discord

In this test, the data analysis technique used was the t test. Before conducting the parametric test, prerequisite analysis tests were carried out first, namely the normality and homogeneity tests. If the prerequisite testing was met, then the data would proceed to the parametric test, namely the t test.



(c)

Figure 3: Normality test for the (a) computational thinking skill, (b) achievement test, and (c) problem solving skill

The normality of the data distribution was tested using Shapiro-Wilk. The distribution of the post-test mean score for the problem solving test was not statistically significantly different from the normal population [$p > 0.05$, $W = 102.69$]. The distribution of the post-test mean scores for the computational thinking test was also not statistically significantly different from the normal population [$p > 0.05$, $W = 64.53$]. The distribution of the post-test mean score for the achievement test was also not statistically significantly different from the normal population [$p > 0.05$, $W = 87.07$]. Therefore, the distribution of the sample data was assumed to be normal and was used in further statistical analysis using parametric tests.

Table 3: Homogeneity test

Variables	Levene's test	p value
Problem solving	0,627	0,472
Computational thinking	0,506	0,614
Achievement test	0,612	0,671

After the normality test, the data was tested for homogeneity. In this study, Levene's Test was used to compare the pre-test data for both groups for each variable. The variance of the pre-test for problem solving, computational thinking and achievement test scores showed that statistically, there was no significant difference (p> 0.05) between the experimental group and the control group. This assumption can be used to conduct the t-test.

Table 4: Results of the T-test

Variables	Groups	N	Mean differences	t	df	P
Posttest						
			Independent sample T-test			
Variables	**Groups**	**N**	**Mean differences**	**t**	**df**	**P**
Problem solving skill	Experimental	32	91,58	12,764	64	0,000
	Control	35	65,25			
Achievement test	Experimental	32	85,85	8,528	64	0,000
	Control	35	63,84			
Computational thinking skill	Experimental	32	75,64	15,886	64	0,000
	Control	35	60,95			
Baseline test (pre-test)						
			Independent sample T-test			
Variables	**Groups**	**N**	**Mean differences**	**t**	**df**	**P**
Problem solving skill	Experimental	32	63,64	2,443	64	1,423
	Control	35	63,54			
Achievement test	Experimental	32	62,43	1,333	64	2,232
	Control	35	62,66			
Computational thinking skill	Experimental	32	60,42	2,133	64	2,328
	Control	35	60,32			

Table 4 shows the significant differences of the pretest for problem solving skill [df=64, t=2.443, p-value=0.00, p>0.05], achievement test [df=64, t=1.333, p-value=0.00, p>0.05] and computational thinking skill [df=64, t=2.133, p-value=0.00, p>0.05] between the experimental and control groups. This means that both groups were truly equivalent at the start of the study. The results of the post-test for the problem solving skill [df=64, t=12.764, p-value=0.00, p<0.05], achievement test [df=64, t=8.528, p-value=0.00, p<0.05] and computational thinking skill [df=64, t=15.886, p-value=0.00, p<0.05] between the experimental and control groups are as indicated. They show that the experimental group outperformed the control group in terms of improving their problem solving skill, achievement test and computational thinking skill. As a result, the impact of the treatment was seen in the post-test score after the experimental group applied Flowgorithm and Discord. This can be seen from the p value, where if the p value was smaller than 0.05, the null hypothesis would be rejected. Based on the statistics shown above, H0 was rejected, indicating that there was a difference in post-test scores between the experimental and control groups. It is evident that the application of Flowgorithm and Discord in learning can have a positive influence on the students' problem-solving skill, achievement test, and computational thinking skills.

Overall, the user experience shows that the combination of Flowgorithm and Discord makes a positive contribution to the learning of algorithms and programming. Flowgorithm helps students better understand and apply algorithm concepts through intuitive visualization, while Discord supports effective collaboration and discussion. The integration of these two technologies not only improves the students' understanding of the material but also increases their engagement and motivation in learning, enriching the learning experience and developing deeper computational thinking. The use of Flowgorithm and Discord in this study demonstrates some important implications in the context of learning algorithms and programming. The students' positive experiences with Flowgorithm suggest that the graphical visualization of algorithms helps overcome some of the key challenges of learning programming, such as understanding the logic flow and structure of complex algorithms. The students feel more confident completing programming tasks because they can see a visual representation of the steps they take, which makes it easier for them to identify and correct errors.

4.4 Correlation Analysis of Computational Thinking Skill

The data was collected from two groups of students: a treatment group using Flowgorithm and Discord, and a control group using traditional learning methods. The students' computational thinking skills were measured before and after the intervention using validated tests, and the test results were analyzed using Pearson correlation analysis.

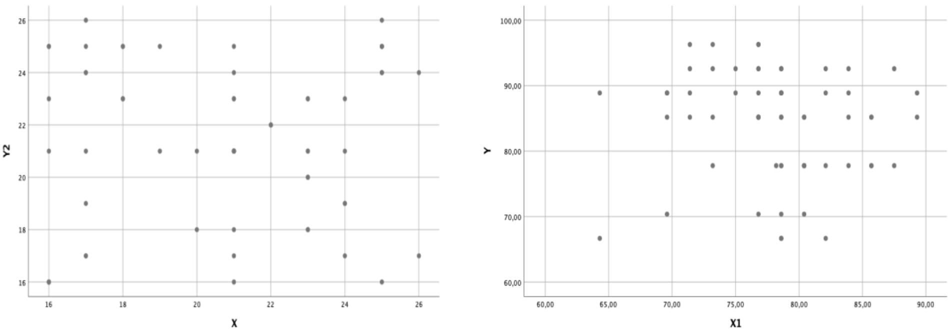


Figure 4: Linearity test results

Before conducting the correlation test, a pre-requisite test was carried out first, namely the linearity test. Based on Figure 4, it can be seen that the points are spread evenly on the diagram, and this means that the three variables are linear. So, it can be concluded that this data can continue to be analyzed using the multiple linear correlation test.

The analysis results showed that there was a significant positive correlation between the use of Flowgorithm and the improvement of the students' computational thinking skills, with a correlation value (r) of 0.756 ($p < 0.01$). This indicates that the more frequently students use Flowgorithm to visualize and understand algorithms, the better their computational thinking skills. In addition, a significant positive correlation was also found between the use of Discord and the improvement of the achievement test, with a correlation value (r) of 0.819 ($p < 0.01$). These results suggest that the collaboration and discussion facilitated by Discord contributed to the development of the students' computational thinking skills.

Table 5: Correlation analysis of flowgorithm and discord with computational thinking skill

		CT	AT	PS	FD
CT	Pearson correlation	1	.030	.048	.756**
	Sig. (2-tailed)		.835	.740	. 000
	N	67	67	67	67
AT	Pearson correlation	.030	1	.432**	.819**
	Sig. (2-tailed)	.835		.002	.000
	N	67	67	67	67
PS	Pearson correlation	.048	.432**	1	.798**
	Sig. (2-tailed)	.740	.002		.004
	N	67	67	67	67
FD	Pearson correlation	.756**	.819**	.798**	1
	Sig. (2-tailed)	. 000	.000	.004	
	N	51	51	51	51
**. Correlation is significant at the 0.01 level (2-tailed)					

When analyzing the combined use of Flowgorithm and Discord, a stronger correlation with improved problem-solving skills was found, with a correlation value (r) of 0.798 ($p < 0.01$). This suggests that the integration of these two technologies provides a stronger synergistic effect than the use of each technology separately. The results of this correlation analysis indicate that the use of Flowgorithm and Discord not only helps students understand the concepts of algorithms and programming but also significantly improves their computational thinking skills. Flowgorithm helps students visualize and tackle problems logically and systematically, while Discord provides a platform for collaboration and discussion that deepens their understanding and application of the concepts.

The findings provide empirical evidence that there is a significant positive relationship between the use of Flowgorithm and Discord, and the development of the students' computational thinking skills. The integration of these two technologies can strengthen the learning of algorithms and programming and help the students develop skills that are essential in computational problem solving. These results provide valuable insights for educators in designing effective curriculum by utilizing technology to enhance the students' computational skills. Furthermore, this study also explored aspects of the students' experiences using Flowgorithm and Discord in learning algorithms and programming. Through questionnaires, the students reported that the visualizations provided by Flowgorithm helped them understand the algorithm steps more clearly and systematically. They felt more confident constructing algorithms because they could see a visual representation of the logic they were creating. This not only improved their conceptual understanding but also motivated them to delve deeper into the material.

5. Discussion

After the data analysis, it was now safe to record that the use of Flowgorithm and Discord in teaching algorithms and programming proved effective at reducing the error rate in the students' programming assignments. The experimental class showed a lower error rate, indicating that algorithm visualization and online collaboration play an important role in the understanding and correct application of algorithm concepts. These findings support the hypothesis that these technologies can enhance algorithm and programming learning and develop the students' computational thinking better compared to traditional learning methods.

The findings of this study are in line with those of Ayob et al. (2022), who found that algorithm visualization tools help students understand and remember complex concepts more efficiently than traditional methods. Tools such as Flowgorithm have been shown to reduce programming error rates by making it easier to visualize the algorithm steps. The study by Robinson (2023) showed that active visualization increased student engagement and concept understanding. On the collaboration side, studies by Gupta (2023) and Robinson (2022) showed that online collaboration platforms such as Discord and Slack improved student communication and engagement. Odinokaya et al. (2021) noted that asynchronous communication through digital platforms can strengthen learning, while Anwar et al. showed that collaboration platforms help by enhancing a sense of community and support among students.

Thus, the findings of this study reinforce the findings of previous research on the benefits of visualization tools and collaboration platforms in programming education (Antonietti et al., 2022). However, by combining Flowgorithm and Discord, this research shows that the integration of these two technologies can provide significant additional benefits, such as reduced error rates, improved understanding of concepts, and better code quality. In addition, this study has revealed that positive user experiences and increased student engagement are also important factors in successful learning. The findings provide practical guidance for educators to help them adopt these technologies in their curriculum, as well as paving the way for further research into the use of combined technologies in computing education.

Several previous studies show that visualization tools such as Flowgorithm can help students understand algorithm concepts better and reduce programming error rates (Divayana et al., 2021). In addition, research on the use of online collaboration platforms such as Discord has also shown positive results, improving material comprehension and academic success. This indicates that combining visualization technologies such as Flowgorithm with collaboration platforms such as Discord can significantly improve the effectiveness of learning algorithms and programming. It provides practical guidance for educators to adopt these technologies in their teaching, creating a more dynamic, interactive learning environment and supporting the development of the students' computational competencies (Huda et al., 2019; Waskito et al., 2023). The findings also pave the way for further research on the use of other innovative technologies in computing education to continuously improve the quality and effectiveness of learning in this increasingly important field.

The results of this study have confirmed these findings, showing that the use of Flowgorithm significantly reduced the students' error rate (from 35% in the control group to 15% in the treatment group) as well as increased the students' engagement and motivation when learning. These studies confirm that the use of Discord in the learning of algorithms and programming improves collaboration, reduces student frustration, and accelerates problem solving by providing real-time assistance, which in turn improves the quality of learning (Wulansari et al., 2023). Furthermore, this study found that the integration of Flowgorithm and Discord provided a stronger synergistic effect than the use of these technologies separately.

The correlation analysis showed there to be a stronger relationship between the combined use of these technologies and the improvement of the students' computational thinking skills. This study also added a qualitative dimension, indicating that the students felt more motivated and helped by the visualization and collaboration provided by the two technologies, resulting in a more enjoyable and supportive learning experience (Aslan, 2021). Thus, this study strengthens and extends the findings of the previous research, providing empirical evidence that the combination of Flowgorithm and Discord can significantly improve student learning outcomes. These results provide practical guidance for educators in designing effective curriculum utilizing technology to improve the students' computational skills. Overall, this study confirms that technological innovation in education has the potential to significantly improve student skills and knowledge.

Discord, as a collaboration platform, also provides significant benefits when it comes to learning. Students report that they find it easier to overcome the difficulties encountered in programming assignments because they can discuss and exchange ideas with their classmates and instructors in real-time (Odinokaya et al., 2021). The screen sharing feature and ability to provide immediate feedback are helpful in problem solving, making the learning process more interactive and supportive. The discussions that take place in study groups on Discord also help deepen the students' understanding of the material as they get to hear different perspectives and approaches to problem solving. In addition, the use of Discord helps to create a more inclusive and collaborative learning environment (Novalinda et al., 2023; Robinson, 2022). Students who may feel awkward or hesitant to ask questions in a physical classroom feel more comfortable participating in online discussions. This increases the active participation of all students, not just those who are usually more vocal. As a result, learning becomes more dynamic and interactive, which contributes to the overall improvement of computational thinking skills.

Flowgorithm and Discord have had a huge impact on developing computational thinking skills in complementary ways. Flowgorithm, as a software for creating flowcharts, helps users break down complex problems into simple steps (decomposition), identify recurring patterns (pattern recognition), and simplify problems by focusing on important elements (abstraction). It also allows users to design algorithms in a structured manner and test their logic before implementing them in code (Gupta, 2023; Wulansari et al., 2023). Discord, as a communication platform, facilitates real-time discussions and collaboration between users, enabling the exchange of ideas, direct guidance from mentors, and questions and answers that speed up the learning process. By using Flowgorithm and Discord together, users can design and visualize algorithms in Flowgorithm, discuss them in Discord for feedback, and debug and adjust the code with community support. This combination supports the development of comprehensive computational thinking skills, including the analytical, technical and collaborative aspects, which are essential for effective and efficient computing problem solving.

6. Conclusion

This study is that the incorporation of Flowgorithm and Discord into the learning of algorithms and programming can increase student computational thinking. This research has significant theoretical and practical implications. Theoretically, this research could enrich the literature on teaching algorithms and programming by exploring how the combination of visualization tools such as Flowgorithm and collaboration platforms such as Discord can improve the conceptual understanding and computational thinking skills. Practically, the findings from this research can provide guidance for educators in designing more effective and interactive curricula and teaching methods. The combination of Flowgorithm and Discord can facilitate more collaborative learning, support students in understanding the material better, and improve the critical and analytical thinking skills needed in the world of computing. It can also help educational institutions adopt appropriate technology to improve the quality of learning and student learning outcomes.

This research contributes to innovation in learning algorithms and programming. This approach can serve as a model for educators to enable them to create more dynamic and relevant learning experiences for students in the digital age. Further research needs to be conducted to test its applicability across different thinking skill and learning contexts. Based on the findings, for teachers, it is recommended to integrate Flowgorithm and Discord into their teaching methods, utilizing Flowgorithm help students visualize the algorithms while using Discord to support real-time discussions, collaboration and guidance. For curriculum designers, it is recommended to incorporate the use of Flowgorithm and Discord into the Algorithms and Programming curriculum structure, ensuring that these tools are used sustainably and integrated with clear learning objectives. For students, it is recommended to actively utilize Flowgorithm to break down and visualize programming problems as well as encourage them to actively participate in discussions and collaborations on Discord to deepen their understanding and computational thinking skills.

7. References

Antonietti, C., Cattaneo, A., & Amenduni, F. (2022). Can teachers' digital competence influence technology acceptance in vocational education? *Computers in Human Behavior*, *132*, Article 107266. https://doi.org/10.1016/j.chb.2022.107266

Aslan, A. (2021). Problem-based learning in live online classes: Learning achievement, problem-solving skill, communication skill, and interaction. *Computers & Education*, *171*, Article 104237. https://doi.org/10.1016/j.compedu.2021.104237

Ayob, M. A., Hadi, N. A., Pahroraji, M. E. H. M., Ismail, B., & Saaid, M. N. F. (2022). Promoting 'discord' as a platform for learning engagement during Covid-19 pandemic. *Asian Journal of University Education*, *18*(3), 663–673. https://doi.org/10.24191/ajue.v18i3.18953

Bernacki, M. L., Greene, J. A., & Crompton, H. (2020). Mobile technology, learning, and achievement: Advances in understanding and measuring the role of mobile technology in education. *Contemporary Educational Psychology*, *60*, Article 101827. https://doi.org/10.1016/j.cedpsych.2019.101827

Cansu, F. K., & Cansu, S. K. (2019). An overview of computational thinking. *International Journal of Computer Science Education in Schools*, *3*(1), 17–30. https://doi.org/10.21585/ijcses.v3i1.53

Chen, C. Y., Chen, P. C., & Chen, P. Y. (2013). Teaching quality in higher education: An introductory review on a process-oriented teaching-quality model. *Total Quality Management and Business Excellence*, *25*(1–2), 36–56. https://doi.org/10.1080/14783363.2011.637789

Christensen, L. B. (2001). *Experimental methodology* (8th ed.). Allyn and Bacon.

Christensen, R., & Knezek, G. (2016). *Relationship of mobile learning readiness to teacher proficiency in classroom technology integration* [Conference session]. 13th International Conference on Cognition and Exploratory Learning in Digital Age (pp 203–306).

Deveci Topal, A., Çoban Budak, E., & Kolburan Geçer, A. (2017). The effect of algorithm teaching on the problem-solving skills of deaf-hard hearing students. *Program: Electronic Library and Information Systems*, *51*(4), 354–372. https://doi.org/10.1108/PROG-05-2017-0038

Divayana, D. G. H., Suyasa, P. W. A., & Widiartini, N. K. (2021). An innovative model as evaluation model for information technology-based learning at ICT vocational schools. *Heliyon*, *7*(2), e06347. https://doi.org/10.1016/j.heliyon.2021.e06347

Ezeamuzie, N. O., Leung, J. S. C., Garcia, R. C. C., & Ting, F. S. T. (2022). Discovering computational thinking in everyday problem solving: A multiple case study of route planning. *Journal of Computer Assisted Learning*, *38*(6), 1779–1796. https://doi.org/10.1111/jcal.12720

Florea, C., David, D., & Pop, A. (2010). An approach to the didactic activity involving the use of new information and communication technology. *Procedia – Social and Behavioral Sciences*, *2*(2), 1699–1702. https://doi.org/10.1016/j.sbspro.2010.03.968

Gao, X., & Hew, K. F. (2022). Toward a 5E-based flipped classroom model for teaching computational thinking in elementary school: Effects on student computational thinking and problem-solving performance. *Journal of Educational Computing Research*, *60*(2), 512–543. https://doi.org/10.1177/07356331211037757

Gong, D., Yang, H. H., & Cai, J. (2020). Exploring the key influencing factors on college students' computational thinking skills through flipped-classroom instruction. *International Journal of Educational Technology in Higher Education*, *17*(1), Article 19. https://doi.org/10.1186/s41239-020-00196-0

Graham, L. J., White, S. L. J., Cologon, K., & Pianta, R. C. (2020). Do teachers' years of experience make a difference in the quality of teaching? *Teaching and Teacher Education*, *96*, Article 103190. https://doi.org/10.1016/j.tate.2020.103190

Gupta, A. (2023). Diversity, dignity, equity, and inclusion in the age of division, discord and disunion: Stereotyping, sexist, hegemony in education. *International Education Studies*, *16*(1), 110–117. https://doi.org/10.5539/ies.v16n1p110

Huda, A., Azhar, N., Almasri, & Fadli. (2019). Design of learning media graphic design through Android-technology based. *International Journal of Recent Technology and Engineering (IJRTE)*, *8*(5), 254–258.

Lai, C., & Jin, T. (2021). Teacher professional identity and the nature of technology integration. *Computers & Education*, *175*, Article 104314. https://doi.org/10.1016/j.compedu.2021.104314

Lai, C.-F., Zhong, H.-X., & Chiu, P.-S. (2021). Investigating the impact of a flipped programming course using the DT-CDIO approach. *Computers & Education*, *173*, Article 104287. https://doi.org/10.1016/j.compedu.2021.104287

Lee, N. (2013). *A conceptual framework for technology-enhanced problem-based learning in construction engineering and management education* [Conference session]. 2013 ASEE Annual Conference & Exposition, Atlanta, Georgia. ASEE. https://doi.org/10.18260/1-2--19049

Lemay, D., Basnet, R. B., Doleck, T., Bazelais, P., & Saxena, A. (2021). Instructional interventions for computational thinking: Examining the link between computational thinking and academic performance. *Computers and Education Open*, *2*, Article 100056. https://doi.org/10.1016/j.caeo.2021.100056

Li, C., Ip, H. H. S., Wong, Y. M., & Lam, W. S. (2020). An empirical study on using virtual reality for enhancing the youth's intercultural sensitivity in Hong Kong. *Journal of Computer Assisted Learning*, *36*(5), 625–635. https://doi.org/10.1111/jcal.12432

Liao, C. H., Chiang, C.-T., Chen, I.-C., & Parker, K. R. (2022). Exploring the relationship between computational thinking and learning satisfaction for non-STEM college students. *International Journal of Educational Technology in Higher Education*, *19*(1), Article 43. https://doi.org/10.1186/s41239-022-00347-5

Lohr, S. L. (2010). *Sampling: Design and analysis* (2nd. ed.). Cengage Learning.

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, *11*(1), 1–17. https://doi.org/10.1080/20004508.2019.1627844

Novalinda, R., Giatman, M., Syahril, Wulansari, R. E., & Trung Tin, C. (2023). Constructivist computer-based instruction (CBI) approach: A CBI flipped learning integrated problem based and case method (PBL-cflip) in clinical refraction course. *International Journal of Online and Biomedical Engineering (IJOE)*, *19*(05), 42−56. https://doi.org/10.3991/ijoe.v19i05.37707

Odinokaya, M. A., Krylova, E. A., Rubtsova, A. V., & Almazova, N. I. (2021). Using the Discord application to facilitate EFL vocabulary acquisition. *Education Sciences*, *11*(9), Article 470. https://doi.org/10.3390/educsci11090470

Peters-Burton, E. E., Cleary, T. J., & Kitsantas, A. (2018). Computational thinking in the context of science and engineering practices: A self-regulated learning approach. In D. Sampson, D. Ifenthaler, J. Spector, & P. Isaías (Eds.), *Digital technologies: Sustainable innovations for improving teaching and learning* (pp. 223–240). Springer International Publishing. https://doi.org/10.1007/978-3-319-73417-0_13

Robinson, B. (2022). "yeet nitro boosted": A postdigital perspective on young people's literacy engagements with the Discord platform. *Literacy Research: Theory, Method, and Practice*, *71*(1), 359–376. https://doi.org/10.1177/23813377221115738

Robinson, B. (2023). Governance on, with, behind, and beyond the Discord platform: A study of platform practices in an informal learning context. *Learning, Media and Technology*, *48*(1), 81–94. https://doi.org/10.1080/17439884.2022.2052312

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, *14*(1), Article 42. https://doi.org/10.1186/s41239-017-0080-z

Waskito, W., Wulansari, R. E., Syahri, B., Erizon, N., Purwantono, P., Yufrizal, Y., & Tee, T. K. (2023). Countenance evaluation of virtual reality (VR) implementation in machining technology courses. *Journal of Applied Engineering and Technological Science (JAETS)*, *4*(2), 825–836. https://doi.org/10.37385/jaets.v4i2.1917

Wulansari, R. E., Dewi, S. M., Marta, R., Sakti, R. H., Primawati, & Tin, C. T. (2023). *Designing the computer assisted instruction (CAI) integrated case method-flipped classroom on engineering education* [Conference session]. Proceedings of the 9th International Conference on Technical and Vocational Education and Training (ICTVET 2022) (pp. 198–205). https://doi.org/10.2991/978-2-38476-050-3_22

Wulansari, R. E., Marta, R., Sakti, R. H., Dewi, S. M., Safitri, D., Kuralbayevna Kassymova, G., Folkourng, F., & Kumar, V. (2023). Computer assisted instruction (CAI) integrated case method-flipped classroom: Innovative instructional model to improve problem-solving skill and learning outcome of TVET students. *Journal of Technical Education and Training*, *15*(4), 103–116. https://doi.org/10.30880/jtet.2023.15.04.009