Education and Information Technologies
https://doi.org/10.1007/s10639-023-11617-8

Effect of software development course on programming self-efficacy

Attila Kovari[1,2,3] · **Jozsef Katona**[4,5]

Received: 22 December 2022 / Accepted: 19 January 2023

Abstract
Negative attitudes and perceptions on programming impair the effectiveness of learning programming skills. In this study the attitude related to programming, problem solving, and self-views on importance of IT/programming knowledge were assessed by pre- and post-test completed at the beginning and at the end of a software development course. The study was conducted using an online questionnaire and four different dimensions were measured by a survey consisting 23 items. The results show positive moderate associations between self-commitment in problem solving and algorithmic and problem solving ability and negative weak relationship with lack of self-confidence in programming. K-means algorithm showed that the students could be classified into two main groups stronger and weaker self-confidence in programming. In the case of both clusters, it was possible to achieve a positive change in attitudes related to programming. In the case of weaker self-confidence in programming, a greater change can be observed in the attitudes, which can be considered an important result from the point of view of the effectiveness of the software development course. The research presented in the article proves that attitudes related to programming can be influenced in a positive direction both in the case of those with stronger, but even more so in the case of those with weaker attitudes.

Keywords Programming attitude · Self-efficacy · Self-confidence · Self-assessment · Pre- post-test · Clustering

1 Introduction

In the field of programming education, several non-cognitive factors such as self-awareness, self-confidence and self-efficacy can be associated with learning outcomes (Alhadabi & Karpinski, 2020). Furthermore, researches (Liu et al, 2021)

✉ Attila Kovari
kovari.attila@uni-eszterhazy.hu

Extended author information available on the last page of the article

Published online: 09 February 2023

revealed that a correlation can be shown between self-confidence and self-efficacy. People with higher self-confidence are able to manage different situations efficiently and independently (Kalita, 2021). According to Bandura (2010) motivation, reinforcement, and previous experiences increase an individual's self-confidence. Self-confidence is defined in such a way that the individual believes himself to be capable, significant and worthy of solving a task. According to Ozuorcun and Tabak (2012) the computer activities increase self-confidence and encourage students to be motivated.

Self-efficacy refers to the ability to solve a problem or task, thus being able to organize and carry out the activities necessary to achieve the given results (Zwart et al, 2020). Self-efficacy is the confidence in one's own abilities, which can be used to perform a novel or difficult task, and it is closely related to motivation. Overall, self-efficacy is a person's belief in their ability to achieve success. Studies show that task based education may has a positive significant effect on self-efficacy (Garaika et al, 2019). Self-efficacy is related to learning and achievement and adaptability to new technology. Self-efficacy is the key to success in any effort, including learning to program.

Students with high self-efficacy have higher expectations of themselves, devote more work to overcoming difficulties, and thus are more successful in tackling the more complex problems that arise (Bandura, 2012). Self-efficacy beliefs influence whether individuals think in self-enhancing or self-debilitating ways and how motivated they are and how well they persevere in the face of adversity (Schwarzer & Warner, 2013).

In the field of programming, self-efficacy combines the ability to solve programming problems independently, the self-confidence, self-confidence and motivation required for this. Some individual factors contribute to student engagement, such as outcomes, motivation, and goals (Eltegani & Butgereit, 2015) (Yilmaz & Koc 2021). In many cases, students face difficulties in solving complex programming and algorithmization problems, and students with low self-efficacy also fail because they do not struggle and do not invest enough energy to overcome these difficulties.

If examine the question from another point of view and focus on learning, Bloom's Taxonomy (Bloom, 1984) or its revised version (Anderson and Krathwohl 2001) primarily helps to interpret the learning processes and consequently the assessment of this process. The taxonomy describes knowledge hierarchically and cumulatively. The original taxonomy defines three areas of knowledge (cognitive, affective and psychomotor) and divides the areas into six levels. The acquisition and assessment of knowledge is a vital part of the education system and has been studied in several papers. Self-assessment is a good way to improve learning outcomes and also provides motivation for learning (Alaoutinen et al. 2010). Díez-Palomar et al. (2020) came to the conclusion that students who consciously evaluate their own and others' work are more successful. Unfortunately, students who do not have adequate motivation, self-efficacy, and perseverance in solving problems are unable to cope with the difficulties that can lead many to demotivation (Gomes & Mendes, 2014). In some cases, it is not necessarily the lack of competence or skills that can be blamed for avoiding programming-related training, but an unreasonably low assessment of self-efficacy. Based on the experiences summarized above, research was

conducted in order to be able to analyze how a programming course that expands programming knowledge can influence the attitude related to programming, problem solving, and self-views on future of IT/programming knowledge.

Learning programming can motivate, develop self-confidence and self-esteem, and overcoming the many obstacles and problems students face during their studies can improve their learning experience and effectiveness. Furthermore, in the context of the above, the success of the acquisition of programming also depends on self-efficacy, ie the individual's opinion on his or her own performance. Project tasks based on programming problem solving can increase students' perceptions of self-efficacy by increasing self-confidence, so educational efforts should focus on improving students' self-confidence.

Related to the previous findings, the research presented in this study analyzes the self-confidence in programming, problem solving, and the importance of IT/programming knowledge, as well as the attitude related to the future perspectives of IT/programming knowledge. In addition to all of this, the investigation also covered how the attitudes related to programming can be influenced in the case of students with previously different attitudes. Finally, we make recommendations that can help to develop and adapt the methodological possibilities of courses aimed at learning programming.

In connection with the previous findings, the article analyzes the students' attitude to programming, based on a survey of algorithmic problem solving and future prospects. In order to ensure the validity of the results, we examine the consistency of the questionnaire and the correlations between the individual determined features. We continue by revealing the main clusters students can be classified into based on the self-assessment of programming and using this to evaluate the impact of the software development course on the students' programming self-assessment.

2 Background

The aims of the study to evaluate the attitude related to four main factors: self-commitment in problem solving, self-confidence of algorithmic and problem solving ability, self-confidence in programming, and vision of future and importance of IT/programming knowledge. In accordance with the objectives, we reviewed the related literary background by highlighting two main topics, one is programming and attitudes and the other is problem solving and attitudes. We used the Scopus database to map the relevant studies of the literary background. For the two topics, we have selected the most influential articles of the past 5 years that have received more citations. The two queries were as follows:

TITLE-ABS-KEY (algorithmic AND thinking OR problem AND solving) AND TITLE-ABS-KEY (self-efficacy OR self-confidence OR self-commitment) AND PUBYEAR > 2016 AND (LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE, "cp") OR LIMIT-TO (DOCTYPE, "bk") OR LIMIT-TO (DOCTYPE, "ch")) AND (LIMIT-TO (SUBJAREA, "COMP") OR LIMIT-TO (SUBJAREA, "ENGI")) AND (LIMIT-TO (LANGUAGE, "English")).

TITLE-ABS-KEY (programming OR software AND development) AND TITLE-ABS-KEY (self-efficacy OR self-confidence OR self-commitment) AND PUBYEAR > 2016 AND (LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE, "cp") OR LIMIT-TO (DOCTYPE, "bk") OR LIMIT-TO (DOCTYPE, "ch")) AND (LIMIT-TO (SUBJAREA, "COMP") OR LIMIT-TO (SUBJAREA, "ENGI")) AND (LIMIT-TO (LANGUAGE, "English")).

The articles collected by the search were reviewed and those related and relevant to the topic were organized and and we highlighted the main findings.

3 Background for attitudes on algorithmic thinking and problem solving

E. Y. Ince and M. Koc (2020) presented an experimental study aimed to investigate the cognitive and affective consequences of Young Engineer's Workshop on the development of students' computational thinking (CT) competence. They used a one-group pretest–posttest model within a quasi-experimental design. In their research 17 (grades 5–6) + 15 (grades 9–10) = 32 student participants were included. They used and application form to collect data, a scale for CT, a satisfaction questionnaire, and student diaries. The research results of CT showed a significant increase on algorithmic and critical thinking whereas no significant changes in creativity, cooperation, and problem-solving. The examined test subjects reported high satisfaction and enjoyment of workshop activities, increased interest, and career planning in programming, and improved self-confidence in robotics project development. The paper suggests that teaching programming can be an effective way to foster CT to some extent but not an adequate or complete solution.

Standl and Bodenstein (2021) analyzes the development of self-efficacy in relation to algorithmic thinking and programming. They analyzed whether the students were confident in solving the tasks and examined how difficult they judged the programming task to be. Based on the results of this paper, suggestions related to the planning of learning and education process can be formulated. Jaipal–Jamanin & Angeli (2017) examined elementary preservice teachers' (N = 21) self-efficacy, understanding of science concepts, and computational thinking and the conclusions derives that teachers' interest and self-efficacy with robotics increased.

Fanchamps et al (2021) examines the evolution of algorithmic thinking among elementary school students. In the analysis, the development process of the Sense-Reason-Act (SRA) cycle-based program was examined. The measurement of self-efficacy showed marginal differences based on the comparison of the preliminary and post-tests.

Peel et al (2022) summarize that many teachers, due to the lack of IT and programming experience, are not adequately prepared to effectively integrate algorithmic thinking into technical knowledge (Aljowaed & Alebaikan, 2018; Sands et al., 2018; Wu et al., 2018). This inexperience can lead to low self-efficacy and low self-confidence (Aljowaed and Alebaikan, 2018; Rich et al. 2021).

Soykan and Kanbul (2018) analized k12 students' self-efficacy regarding coding. They determined the students' coding self-efficacy levels according to

whether taking a coding course or not. The authors found that the students who take participate coding education had higher self-efficacy compared to students who did not receive coding education. The results showed that it was not significant difference in coding self-efficacy according to gender with neither the students taking the course or not. Similar conclusions were reached by Kukul et al (2017), who used a self-developed scale. Based on the results, it was concluded that similar conclusions can be made not only to secondary schools but also to k12 schools.

4 Background for attitudes on programming and software development

Tsai (2019) examined the effect of visual programming language (VPL) on the effectiveness of understanding basic programming of students with different levels of self-efficacy. The participants taking general courses at a university in southern Taiwan. The results showed that the VPL based teaching of basic programming concepts improve the understanding of basic programming and this effect was better in students with moderate and low self-efficacy.

Shanmugam et al (2019) investigated the concept of Computational Thinking, which aims to improve cognitive ability by solving problems through programming. The study highlights that computational thinking skills play an important role in its development and most countries are aware of the importance of these skills in this century. Emphasis is placed on trying to improve students' understanding and learning motivation by integrating digital technologies, this approach provides an important link in increasing the motivation of the digital generation and the effectiveness of learning. In their study, they examined five motivational aspects, internal goals, external goals, the value of the task, the control of learning beliefs and self-efficacy. Based on the results, it was established that both the Computational Thinking module and the traditional method had a positive effect on the motivation of the students, and improved learning results compared to students using the traditional method.

Kittur (2020) highlighted that the self-efficacy of electrical engineering students with regard to computer programming tasks and self-regulation varies depending on class standing and prior experience in programming. The survey he used contained a total of 31 items. Based on the results, he determined that there was no statistically significant relationship between basic programming tasks and dependence, however, the complex programming tasks and self-regulation proved to be significant based on his class standing and previous programming experience. The paper concluded that better programming experience leads to greater programming self-efficacy in relation to programming tasks. As a result, if the universities provide students with more practical experience, it helps to increase their self-efficacy and self-confidence.

Zhang et al (2021) used a progressive thinking (through experiences, learning by doing, projects etc.) method to develop students' computational thinking skills. In their study conducted with the help of 49 test subjects who took a programming course and they were divided into two groups. During the training, one group received progressive and the other non-progressive education. The results they obtained showed that the progressive teaching method was more effective, and this

group also showed a significant improvement in programming self-efficacy. They also observed that this group of participants were more effective in cooperative learning, critical thinking, and problem-solving skills.

5 Research questions

Thinking further about the research directions revealed in the analyzed articles we formulated a research goal that will help to reveal to what extent practice-oriented programming knowledge increases the participants' attitude to programming and programming self-efficacy. We searched for the answer to how students' attitude to programming changes depending on self-confidence during the development of programming skills that require problem solving. Based on the above, the aims of our study to evaluate the attitude related to four main factors: self-commitment in problem solving, self-confidence of algorithmic and problem solving ability, self-confidence in programming, and vision of future and importance of IT/programming knowledge.

The research presented in this study the attitude related to programming, problem solving, and self-views on importance of IT/programming knowledge were assessed by pre- and post-test completed at the beginning and at the end of the programming course. The main general research questions are formulated as follows:

RQ1What effect does the programming course have on the attitude related to algorithmic and problem solving ability as far as the future importance of IT/programming?
RQ2What relationship can be shown between the four main factors examined in connection with the attitude analysis related to programming?
RQ3Based on the analysis of the four main factors, the students who participated in the research can be classified into how many clusters and with what characteristics can be described for these clusters?

6 Materials and Methods

The test of attitude related to programming, problem solving, and IT vision was conducted in two stages before and after completing a course that developed programming skills. In this course not only the transfer of programming knowledge was realized, but also self-development through programming tasks to be developed independently. It was examined that the improvement of programming knowledge and the establishment of more confident skills in the field of software development will have an impact on self-viewpoints and the perspectives of self-efficacy and self-development in programming and IT knowledge. Self-assessment plays an important role in the development of self-efficacy, the cycle of improvement of self-assessment of programming showing in Fig. 1.

The study was conducted using an online questionnaire at two consecutive different time points in 2022. Both times students filled the questionnaires online. The

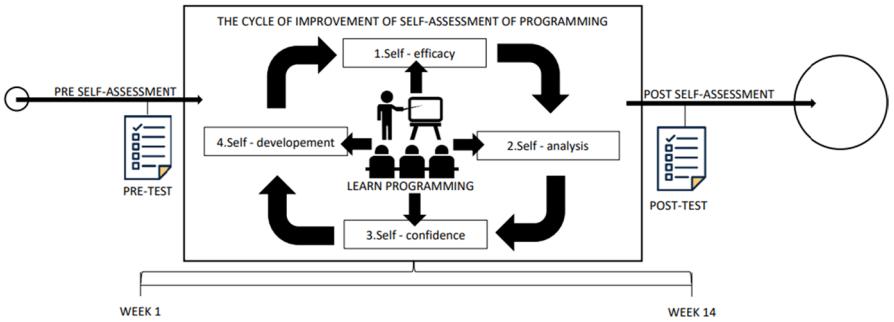

Fig. 1 The cycle of improvement of self-assessment of programming

first survey took place at the beginning of the semester, before the studies of the programming course, it was the pre-test. The second survey took place at the end of the semester after the acquisition of software development and programming knowledge and methods, it was the post-test (Fig. 2).

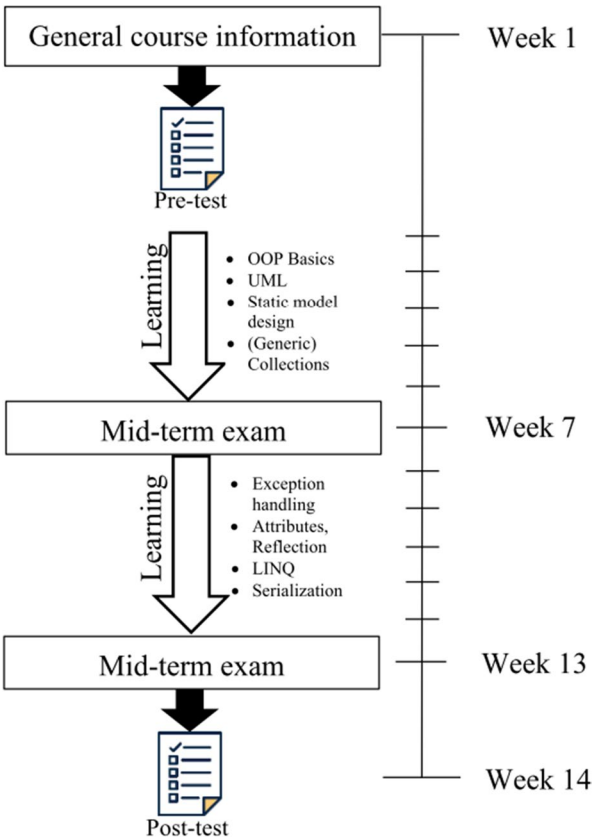

Fig. 2 Schedule of programming course

Table 1 Participants

Participants	Man	Woman	Total
University of Dunaujvaros	50	5	**55**
Sapientia Hungarian University of Transylvania	49	12	**61**
Total	**99**	**17**	**116**

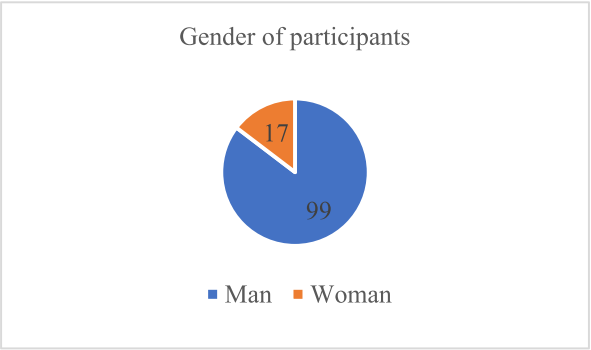

Fig. 3 Genders of participants (N = 116)

6.1 Participants

The participants of the study were 116 IT BSc students with basic programming skills of University of Dunaujvaros, Hungary and Sapientia Hungarian University of Transylvania, Romania who participated in the software development (C# Object-Oriented Programming and UML modelling) courses. The course covering the basic programming knowledge acquired in a previous subject. 55 students completed the questionnaire at University of Dunaujvaros, while 61 students at Sapientia University. Men students were 99 (85%) and women were 17 (15%) total (Table 1 and Fig. 3). The age of the respondents was between 18 and 28 years. Participants did not suffer from a disease affecting mental ability.

6.2 Measures

The examination of attitude towards programming, algorithmic problem solving, and future prospects was carried out by a questionnaire based survey. The evaluation process uses single group pre-test post-test. Four different factors were measured by a survey consisting 23 items:

1. Self-commitment in problem solving (5 items)
2. Self-confidence of algorithmic and problem solving ability (6 items)
3. Lack of self-confidence in programming (5 items)
4. Self-views on future of IT/programming knowledge (7 items)

Table 2 Descriptive statistics

Variables	Pre-test				Post-test			
	Min	Max	Mean	SD	Min	Max	Mean	SD
1. Self-commitment in problem solving	1.20	5.00	3.88	0.84	1.40	5.00	4.11	0.76
2. Self-confidence of algorithmic and problem solving ability	1.83	4.83	3.92	0.72	2.00	5.00	4.18	0.65
3. Lack of self-confidence in programming	1.00	4.40	2.02	1.02	1.00	4.00	1.75	0.88
4. Self-views on future of IT/programming knowledge	2.57	5.00	4.16	0.59	2.43	5.00	4.18	0.57

Table 3 Cronbach's Alpha at pre- and post-tests

Factors	Number of items	Cronbach's Alphas	
		Pre-test	Post-test
1.Self-commitment in problem solving	5	0.83	0.76
2.Self-confidence of algorithmic and problem solving ability	6	0.82	0.79
3.Lack of self-confidence in programming	5	0.92	0.88
4.Self-views on future of IT/programming knowledge	7	0.78	0.76

$N = 116$

The response format for the items was a 5-point Likert scale. The internal consistency of the items was examined by Cronbach's alpha. The Cronbach's alpha measure how closely related a set of items are as a group, this is a measure of scale reliability.

6.3 Procedure

All students were asked to complete an online questionnaire concerning the mentioned fours of 23 items. The pre-test questionnaire was answered at the beginning of the programming course. The post-test was filled in after the completing the teaching period of the course. A total of 116 students completed all items of both tests.

6.4 Analysis

Descriptive statistical analysis of the pre- and post-test results are summarized in Table 2. Mean and standard deviation was calculated for each factors.

Table 3 shows four factors of attitudes towards programming ability and self-views on future of IT knowledge. Cronbach' s alphas of all scales are between 0.76 to 0.92 for the pre- and post-tst respectively. These values meet the criteria indicating reasonable reliability in social science Cronbach alpha > 0.60 (Hair et al., 2006).

Correlations matrix was used to examine the relationships between factors that explain student's attitudes towards confidence in programming based on problem solving, algorithmic ability and self-confidence and views in programming knowledge.

Shapiro–Wilk test was used to check the normality of the distributions. In case of normal distributions, Student's T-test was used for independent samples (for example gender analysis) or related samples (compare pre- and post-test) was used, after Levene test to check the homogeneity of variances. For non-normal distributions Kruskal–Wallis test was used for independent samples and Wilcoxon signed-rank for related samples. The significance level was 0.05.

Following these K-means clustering was used to categorize students into groups based on their confidence in programming ability and knowledge. In addition, the paired-samples T-test was conducted to determine whether the impact of programing course on students' attitudes towards confidence in programming.

7 Results

The next chapters summarize the results of descriptive statistics, reliability analysis, correlation matrix, k-means clustering and dependent paired-samples T-test evaluations.

7.1 Descriptive statistics

Table 2 shows the descriptive statistics of the four factors of the survey.

As can be seen, at the beginning of the semester the students had higher than average attitude in self-commitment in problem solving, algorithmic and problem solving ability, self-views on future of IT knowledge and lower than average for lack of self-confidence in programming.

7.2 Reliability analysis

Table 3 summarize the results of the reliability analysis using internal consistency of the items in case of pre- and post-test. Based on the results, it can be stated that the reliability of the groups of questions characterizing the factors of the questionnaire is adequate.

7.3 Associations between different factors

Correlation matrix shows the associations between factors that explain student's attitudes towards confidence in programming ability and knowledge based on problem solving, algorithmic ability and self-confidence and views in programming knowledge. In order to assess the statistical significance of the correlation, variables should be approximately normally distributed in case of calculating Pearson correlation. If the data are not normally distributed, a Spearman correlation should apply.

Spearman correlation is a nonparametric measure of the strength and direction of association that exists between two variables measured on at least an ordinal scale. Table 4 shows the results of the Shapiro–Wilk normality test and it can be stated that the data are not normally distributed.

Because the variables are not normally distributed Spearman correlations were used to explores the associations between variables. Table 5 summarize the results of Spearman correlations between pre- and post-test factors.

The relationship between the same factors in the pre- and post-tests was very strong (r[15], r[26], r[37], r[48] > 0.9, $p = 0.01$), students responded with similar orientations for the factors for pre- and post-test, as expected. The pre-test results show positive moderate associations between self-commitment in problem solving and algorithmic and problem solving ability (r[12] = 0.548, $p = 0.01$) and self-views on future of IT knowledge (r[14] = 0.404, $p = 0.01$) on the other hand negative weak relationship with lack of self-confidence in programming (r[13] = -0.329, $p = 0.01$). Between self-confidence of algorithmic, problem solving ability and self-views on future of IT/programming knowledge have also positive moderate associations (r[24] = 0.488, $p = 0.01$) while a lack of self-confidence in programming and self-views on future of IT/programming knowledge has negative weak relationship (r[34] = -0.309, $p = 0.01$). A similar can be found in the post-test results. The greatest correlation was for algorithmic and problem solving ability and for lack of self-confidence in programming (r[23] = -0.835, $p = 0.01$).

7.4 Clustering students based on self-assessment of programming

K-means clustering method was used to create groups of students based on the four main factors. The standardized values (Zscore) were the input data of the K-means method and the maximum iterations was 10. Convergence achieved in 5 iterations and the number of cases in each cluster and final cluster centers can be seen in Table 6 & 7 and the results of ANOVA shoving a significant difference between the clusters (groups) (Table 8).

As shown in Fig. 4, the characteristics of the two cluster (group) factors determined by the K-means algorithm differ significantly from each other. Cluster 1 (N = 84) is the group of students with Stronger self-confidence in programming and the Cluster 2 (N = 32) is the group of Weaker self-confidence in programming. The

Table 4 Shapiro–Wilk test of normality

Variable	Pre-test		Post-test	
	W-value	*p*-value	*W*-value	*p*-value
1. Self-commitment in problem solving	0.906	< 0.001*	0.885	< 0.001*
2. Self-confidence of algorithmic and problem solving ability	0.905	< 0.001*	0.902	< 0.001*
3. Lack of self-confidence in programming	0.864	< 0.001*	0.816	< 0.001*
4. Self-views on future of IT/programming knowledge	0.930	< 0.001*	0.932	< 0.001*

*Since the p-value is less than 0.05, the data does not appear to be normally distributed

Table 5 Correlation matrix in case of pre- and post-tests

		Pre-test				Post-test			
	Variable	1	2	3	4	5	6	7	8
Pre-test	1. Self-commitment in problem solving	1							
	2. Self-confidence of algorithmic and problem solving ability	0.548**	1						
	3. Lack of self-confidence in programming	-0.329**	-0.835**	1					
	4. Self-views on future of IT/programming knowledge	0.404**	0.488**	-0.309**	1				
Post-test	5. Self-commitment in problem solving	0.966**	0.541**	-0.334**	0.389**	1			
	6. Self-confidence of algorithmic and problem solving ability	0.518**	0.949**	-0.779**	0.473**	0.514**	1		
	7. Lack of self-confidence in programming	-0.364**	-0.856**	0.961**	-0.279**	-0.369**	-0.818**	1	
	8. Self-views on future of IT/programming knowledge	0.389**	0.428**	-0.265**	0.978**	0.370**	0.423**	-0.223*	1

$N = 116$ students

** Correlation is significant at the 0.01 level (2-tailed)

* Correlation is significant at the 0.05 level (2-tailed)

Table 6 Number of Cases in each Cluster

Cluster	1	84.000
	2	32.000
Valid		116.000
Missing		0.000

Table 7 Final Cluster Centers

Pre-Test	Cluster	
	1	2
Zscore: Self-commitment in problem solving	0.298	-0.782
Zscore: Self-confidence of algorithmic and problem solving ability	0.496	-1.303
Zscore: Lack of self-confidence in programming	-0.475	1.246
Zscore: Self-views on future of IT/programming knowledge	0.332	-0.872

group of stronger self-confidence has positive self-commitment in problem solving, self-confidence of algorithmic and problem solving ability and self-views on future of IT/programming knowledge. On the other hand this group have negative lack of self-confidence in programming. These factors characterize those who have stronger self-confidence in programming. The Cluster 2 is the opposite of Cluster 1 so this group has weaker self-confidence in programming.

As it can be seen in Table 9 that means of post-test shows better results than pre-test for the main four factors in case of Cluster 1 and 2 too. Cluster 1 has means 4.13 in case of pre-test and 4.34 for post-test, meanwhile students with Cluster 2 has means 3.22 in case of pre-test and 3.50 for post-test regarding self-commitment in problem solving. Cluster 1 has means 4.27 concerning pre-test and 4.5 in case of post-test while Cluster 2 has means 2.98 for pre-test and 3.38 in case of post-test regarding Self-confidence of algorithmic and problem solving ability. In contrast with self-views on future of IT/programming knowledge where the post-test has the same means in pre- and post-test in case of Cluster 1 and the means has changed from 3.65 to 3.73 for Cluster 2. In contrary, the means shows decrease from 1.54 to 1.33 in lack of self-confidence in programming for pre-test in case of Cluster 1 and from 3.29 to 2.87 concerning post-test regarding Cluster 2.

7.5 The impact of software development course on students' self-assessment of programming

The related samples Wilcoxon signed-rank test was used to determine whether the impact of software development course on students' self-assessment of programming. The results of the pre- and post-tests for all students are summarized in Table 10.

Wilcoxon signed-rank test show that software development course had a statistically significant positive effect on average ratings for self-commitment in problem

Table 8 Results of ANOVA for the Final Clusters

Pre-Test	Cluster		Error		F	Sig
	Mean Square	df	Mean Square	df		
Zscore: Self-commitment in problem solving	27.045	1	0.772	114	35.054	0.000
Zscore: Self-confidence of algorithmic and problem solving ability	75.042	1	0.351	114	214.091	0.000
Zscore: Lack of self-confidence in programming	68.606	1	0.407	114	168.579	0.000
Zscore: Self-views on future of IT/programming knowledge	33.595	1	0.714	114	47.046	0.000

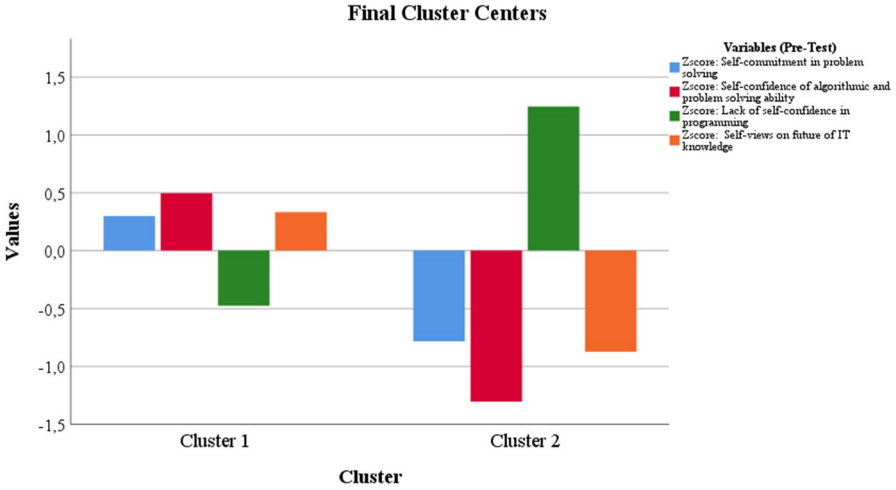

Fig. 4 Bar graph of Final Cluster Centers for Cluster 1 (stronger self-confidence in programming) and Cluster 2 (weaker self-confidence in programming)

solving (Z = 7.917, $p < 0.001$), self-confidence of algorithmic and problem solving ability (Z = 8.417, $p < 0.001$) and self-views on future of IT/programming knowledge (Z = 2.723, $p = 0.006$). In contrary, the test show that software development course had a statistically significant but negative effect on lack of self-confidence in programming (Z = 8.007, $p < 0.001$). The medians of student ratings were 4.0 in case of pre- and 4.2 in case of post-test for self-commitment in problem solving, 4.0 and 4.3 for self-confidence of algorithmic and problem solving ability, 1.8 and 1.4 for lack of self-confidence in programming and finally 4.29 for pre- and post-test also in case of self-views on future of IT/programming knowledge.

The impact of software development course on students' self-assessment of programming students with stronger and weaker self-confidence in programming are summarized in Table 11 and Table 12.

The results show same results for students with stronger and weaker self-confidence in programming. The software development course had a statistically significant positive effect on average ratings for self-commitment in problem solving (Z = 6.71 and Z = 4.301, $p < 0.001$), self-confidence of algorithmic and problem solving ability (Z = 6.912 and Z = 4.82, $p < 0.001$). Self-views on future of IT/programming knowledge is not significant difference in case of students with stronger self-confidence in programming (Z = 1.027, $p = 0.305$) but significant for students with weaker self-confidence in programming (Z = 3–119, $p = 0.002$). In contrary, the test show that software development course had a statistically significant but negative effect on lack of self-confidence in programming in case of students with stronger self-confidence in programming (Z = 6.466 and Z = 4.872, $p < 0.001$). The medians of students with stronger self-confidence in programming ratings were 4.2 in case of pre- and 4.4 in case of post-test for self-commitment in problem solving, 4.33 and 4.5 for self-confidence of algorithmic and problem solving ability, 1.2 and 1.0 for lack of self-confidence in programming and finally 4.43 for pre- and post-test

Table 9 Descriptive statistics for the two clusters

Variables	Cluster 1 stronger self-confidence in programming								Cluster 2 weaker self-confidence in programming							
	Pre-test ($N=84$)				Post-test ($N=84$)				Pre-test ($N=32$)				Post-test ($N=32$)			
	Min	Max	Mean	SD	Min	Max	Mean	SD	Min	Max	Mean	SD	Min	Max	Mean	SD
1. Self-commitment in problem solving	2.80	5.00	4.13	0.62	3.00	5.00	4.34	0.52	1.20	4.60	3.22	0.99	1.40	5.00	3.50	0.94
2. Self-confidence of algorithmic and problem solving ability	3.33	4.83	4.27	0.39	3.50	5.00	4.50	0.34	1.83	3.67	2.98	0.50	2.00	4.17	3.34	0.50
3. Lack of self-confidence in programming	1.00	3.00	1.54	0.61	1.00	2.80	1.33	0.46	2.20	4.40	3.29	0.74	1.40	4.00	2.87	0.74
4. Self-views on future of IT/programming knowledge	3.29	5.00	4.36	0.42	3.29	5.00	4.36	0.41	2.57	4.86	3.65	0.65	2.43	5.00	3.73	0.68

Table 10 Results of Wilcoxon signed-rank pre- post-test for all students

Variable	Median		T	Z-score	*p* value	r
	Pre-test	Post-test				
Self-commitment in problem solving	4.000	4.200	0.000	7.917	<0.001*	-0.520**
Self-confidence of algorithmic and problem solving ability	4.000	4.330	23.500	8.417	<0.001*	-0.553**
Lack of self-confidence in programming	1.800	1.400	3570.000	8.007	<0.001*	0.526**
Self-views on future of IT/programming knowledge	4.290	4.290	340.500	2.723	0.006*	0.179**

$N = 116$

*2-tailed

**A large effect ($|r| \geq 0.5$)

Table 11 Results of Wilcoxon signed-rank pre- and post-test for students with stronger self-confidence in programming (Cluster 1)

Variable	Median		T	Z-score	*p* value	r
	Pre-test	Post-test				
Self-commitment in problem solving	4.200	4.400	0.000	6.710	<0.001*	-0.518**
Self-confidence of algorithmic and problem solving ability	4.330	4.500	20.000	6.912	<0.001*	-0.533**
Lack of self-confidence in programming	1.200	1.000	1431.000	6.466	<0.001*	0.499**
Self-views on future of IT/programming knowledge	4.430	4.430	209.500	1.027	0.305	-0.079**

$N = 84$

*2-tailed

**A large effect ($|r| \geq 0.5$)

Table 12 Results of Wilcoxon signed-rank pre- and post-test for students with weaker self-confidence in programming (Cluster 2)

Variable	Median		T	Z-score	*p* value	r
	Pre-test	Post-test				
Self-commitment in problem solving	3.300	3.700	0.000	4.301	< 0.001*	-0.538**
Self-confidence of algorithmic and problem solving ability	3.170	3.415	0.000	4.820	< 0.001*	-0.603**
Lack of self-confidence in programming	3.300	3.000	496.00	4.872	< 0.001*	0.609**
Self-views on future of IT/programming knowledge	3.570	3.640	11.00	3.119	0.002*	-0.390**

$N = 32$

*2-tailed

**A large effect ($|r| \geq 0.5$)

also in case of self-views on future of IT/programming knowledge. The medians of students with weaker self-confidence in programming ratings were typically lower 3.3 in case of pre- and 3.7 in case of post-test for self-commitment in problem solving, 3.17 and 3.415 for self-confidence of algorithmic and problem solving ability, 3.3 and 3.0 for lack of self-confidence in programming and finally 3.57 for pre- and 3.64 for post-test in case of self-views on future of IT/programming knowledge.

The independent (two-sample) Mann–Whitney rank sum test was used to determine whether show a clear distinction between Cluster 1 and Cluster 2 (Table 13 and 14).

The results show same results for pre- and post-test. It can be concluded that Self-commitment in problem solving in the Cluster 1 was statistically significantly higher than Cluster 2 ($U = 582.5$ for pre-test and $U = 581.5$ for post-test, $p < 0.001$) and same as for Self-confidence of algorithmic and problem solving ability ($U = 41.0$ for pre-test and $U = 583.5$ for post-test, $p < 0.001$) and for Self-views on future of IT/programming knowledge ($U = 508.0$ for pre-test and $U = 587.5$ for post-test, $p < 0.001$). Cluster 2 was statistically significantly higher than Cluster 1 for Lack of self-confidence in programming ($U = 110.5$ for pre-test and $U = 121.0$ for post-test, $p < 0.001$).

8 Discussion

The research presented in the paper examined the attitude related to programming, problem solving, and future perspectives of IT/programming knowledge. 116 IT BSc students from Dunaújváros University and Sapientia University participated in the study. The students attended the software development (C# Object-Oriented Programming and UML modeling) course. To analyze the attitude and its change by the skill development provided by the software development course, a 23-item questionnaire along four dimensions/factors were applied in pre- and post-test basis:

Table 13 Results of Mann–Whitney rank sum test of pre-tests for Cluster 1 and Cluster 2

Variable	Median		*U*	*Z*-score	*p* value	r
	Pre-test (Cluster 1)	Pre-test (Cluster 2)				
Self-commitment in problem solving	4.200	3.300	582.500	-4.726	< 0.001*	-0.439**
Self-confidence of algorithmic and problem solving ability	4.330	3.170	41.000	-8.102	< 0.001*	-0.752**
Lack of self-confidence in programming	1.200	3.300	110.500	-7.697	< 0.001*	-0.715**
Self-views on future of IT/programming knowledge	4.430	3.570	508.000	-5.191	< 0.001*	-0.482**

$N = 116$

*2-tailed

**A large effect ($|r| \geq 0.5$)

Table 14 Results of Mann–Whitney rank sum test of post-tests for Cluster 1 and Cluster 2

Variable	Median		*U*	*Z*-score	*p* value	*r*
	Post-test (Cluster 1)	Post-test (Cluster 2)				
Self-commitment in problem solving	4.400	3.700	581.500	-4.737	<0.001*	-0.440**
Self-confidence of algorithmic and problem solving ability	4.500	3.415	583.500	-8.011	<0.001*	-0.744**
Lack of self-confidence in programming	1.000	3.000	121.000	-7.794	<0.001*	-0.724**
Self-views on future of IT/programming knowledge	4.430	3.640	587.500	-4.694	<0.001*	-0.436**

$N = 116$

*2-tailed

**A large effect ($|r| \geq 0.5$)

1. Self-commitment in problem solving (5 items)
2. Self-confidence of algorithmic and problem solving ability (6 items)
3. Lack of self-confidence in programming (5 items)
4. Self-views on future of IT/programming knowledge (7 items)

When interpreting the results, in the case of factors 1, 2 and 4, a larger value means a positive attitude, while in the case of factor 3, a smaller value. The reliability of the questionnaire was confirmed with the help of Cronbach's Alphas in relation to the 4 examined dimensions.

RQ1What effect does the programming course have on the attitude related to algorithmic and problem solving ability as far as the future importance of IT/programming?

Regarding the results, it can be stated that the averages for the individual factors increased for factors 1, 2 and 4, while decreased for factor 3, based on the comparison of the pre- and post-tests. In the case of the pre-test, the average value of self-commitment in problem solving is 3.88, while in the case of the post-test this value increased to 4.11, which means an increase of 5.9%. Regarding the self-confidence of algorithmic and problem solving ability, the average value of the pre-test was 3.92, while in the case of the post-test this value increased to 4.18, which means an increase of 6.6%. However, regarding self-views on future of IT/programming knowledge, the increase was only small, from 4.16 to 4.18, which means 0.5%. A 13.3% decrease can be observed, from 2.02 to 1.75 in the case of factor 3, the lack of self-confidence in programming in relation to the pre- and post-tests. Figure 5 shows the changes in the mean of the pre- and post-tests regarding the 4 factors.

Overall, it can be concluded that the software development course had a positive effect on students attitudes related to programming and problem solving ability, as long as there were no significant changes in the perception of IT's role in the future. This can be attributed to the fact that even at the beginning of the course, the future role and importance of IT was considered significant.

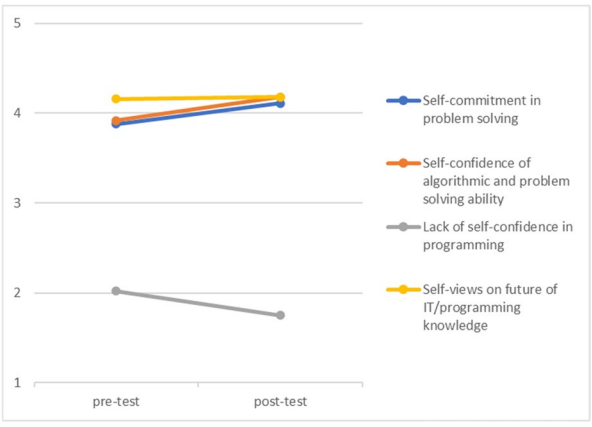

Fig. 5 Changes in the mean of the pre- and post-tests regarding the 4 factors

RQ2What relationship can be shown between the four main factors examined in connection with the attitude analysis related to programming?

The analysis of the relationship between the individual factors showed that the relationship between the same factors in the pre- and post-tests was very strong, students responded with similar orientations for the pre- and post-test, as expected. The pre-test results show positive moderate associations between self-commitment in problem solving and algorithmic and problem solving ability and self-views on future of IT knowledge on the other hand negative weak relationship with lack of self-confidence in programming. Between self-confidence of algorithmic, problem solving ability and self-views on future of IT/programming knowledge have also positive moderate associations while a lack of self-confidence in programming and self-views on future of IT/programming knowledge has negative weak relationship. The greatest correlation between the factors was for algorithmic and problem solving ability and for lack of self-confidence in programming. The results indicate that those who consider themselves better at algorithmic thinking and problem solving also consider themselves more confident in programming, and lower self-esteem appears to a lesser extent. This supports the importance of problem solving and algorithmic thinking in learning and mastering programming. Similar results can be found in the post-test results.

RQ3Based on the analysis of the four main factors, the students who participated in the research can be classified into how many clusters and with what characteristics can be described for these clusters?

The results of clustering using the K-means algorithm on four factors showed that the students could be classified into two main groups. Based on the factor values related to the clusters, Cluster 1 was typical of the group of students who have stronger self-confidence in programming, due to higher self-commitment in problem solving and self-confidence of algorithmic and problem solving ability, as well as lower value for lack of self-confidence in programming. While in the case of Cluster 2, these values were the opposite, which is why this group was named as weaker self-confidence in programming.

Students with stronger self-confidence in programming (Cluster 1) have 5.1% greater ratings in post-test, meanwhile students with weaker self-confidence in programming (Cluster 2) have 8.7% greater ratings in posts-test in case of self-commitment in problem solving. Cluster 1 has 5.4% greater rating in post-test and Cluster 2 has 12.1% greater rating in post-test in case of self-confidence of algorithmic and problem solving ability. In contrast with self-views on future of IT/programming knowledge where the post-test has the same means in pre- and post-test in case of Cluster 1 and 2.2% greater rating in post-test in for Cluster 2. In contrary, the means shows decrease by 13.6% in lack of self-confidence in programming for post-test in case of Cluster 1 and by 12.8% for post-test in case of Cluster 2. Figure 6 shows the changes in the mean of the pre- and post-tests regarding the 4 factors in case of Cluster 1 and Cluster 2.

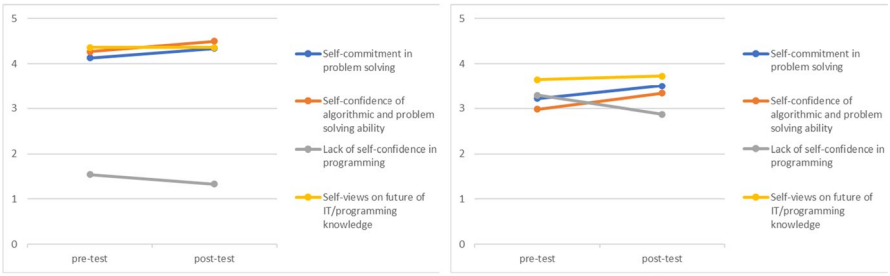

Fig. 6 Changes in the mean of the pre- and post-tests regarding the 4 factors (a: Cluster 1, b: Cluster 2)

In the case of both clusters, it was possible to achieve a positive change in attitudes related to programming. In the case of Cluster 2, i.e. weaker self-confidence in programming, a greater change can be observed, which can be considered an important result from the point of view of the effectiveness of the software development course. Based on the comparison of the dependent, paired-samples tests performed on the entire sample and the independent sample tests performed in the case of the Clusters, it can be concluded that the results of the pre- and post-tests show a significant improvement on the attitude and that these differences also apply to the Clusters are significant. The results support the improvement of the factors examined in connection with the programming attitude as a result of completing the practice-oriented software development course.

9 Conclusions and future work

A low attitude towards programming-related competencies can cause deficiencies in the effective acquisition of programming knowledge. Students' programming performance can be increased if they use methods and techniques that improve their own skills and improve their attitudes and self-efficacy towards programming. First of all, the successful solution of tasks that provide practical knowledge can give you the confidence that can have a positive effect on self-efficacy and attitudes.

The results show that practice-oriented programming knowledge increases participants' attitudes towards programming and programming self-efficacy. The practice-oriented tasks of the programming steps can arouse the curiosity of the students, which can make the programming process, which requires complicated and complex skills, easier and more interesting. All these factors could have improved the students' attitude towards programming. Based on the students' feedback, the practical activities help in understanding the programming paradigms, promote the transparency of the algorithmization process and the development of these skills, which positively affects their opinion of their programming self-efficacy. The research presented in the article proves that attitudes related to programming can be influenced in a positive direction both in the case of those with stronger, but even more so in the case of those with weaker attitudes.

Therefore, the results of the survey will be useful for universities to understand students' confidence in programming and introduce interventions in the curriculum accordingly. In the future, we would like to use the results of the study with the competence map of curricula and subjects related to the field of IT training and in the further development of modern digital methodological possibilities. The feedback provided in the article may be important in the future in the development of both teacher and student attitudes. These results of the survey will be useful for students to reflect on their abilities, as this survey can be used for self-reflection.

Funding Open access funding provided by Eszterhazy Karoly Catholic University.

Data availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest None.

References

Alaoutinen, S., & Smolander, K. (2010, June). Student self-assessment in a programming course using bloom's revised taxonomy. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (pp. 155–159). https://dl.acm.org/doi/10.1145/1822090.1822135

Alhadabi, A., & Karpinski, A. C. (2020). Grit, self-efficacy, achievement orientation goals, and academic performance in University students. *International Journal of Adolescence and Youth, 25*(1), 519–535. https://doi.org/10.1080/02673843.2019.1679202

Aljowaed, M., & Alebaikan, R. A. (2018). Training needs for computer teachers to use and teach computational thinking skills. *Interntional Journal for Research in Education, 42*(3), 237–284.

Anderson, L. W., & Krathwohl, D. R. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. Longman.

Bandura, A. (2010). Self-efficacy. The Corsini encyclopedia of psychology. *John Wiley & Sons, Inc. doi*, *10*(9780470479216), 1–3. https://doi.org/10.1002/9780470479216.corpsy0836

Bandura, A. (2012). Cultivate self-efficacy for personal and organizational effectiveness. *Handbook of Principles of Organizational Behavior: Indispensable Knowledge for Evidence-Based Management*, 179–200. https://doi.org/10.1002/9781119206422.ch10

Bloom, B. S. (1984). *Taxonomy of Educational Objectives*. Allyn and Bacon.

Díez-Palomar, J., García-Carrión, R., Hargreaves, L., & Vieites, M. (2020). Transforming students' attitudes towards learning through the use of successful educational actions. *PloS one*, *15*(10), e0240292. https://doi.org/10.1371/journal.pone.0240292

Eltegani, N., & Butgereit, L. (2015, September). Attributes of students engagement in fundamental programming learning. In *2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE)* (pp. 101–106). IEEE. https://doi.org/10.1109/ICCNEEE.2015.7381438

Erol, O., & Kurt, A. A. (2017). The effects of teaching programming with scratch on pre-service information technology teachers' motivation and achievement. *Computers in Human Behavior, 77*, 11–18. https://doi.org/10.1016/j.chb.2017.08.017

Fanchamps, N. L., Slangen, L., Hennissen, P., & Specht, M. (2021). The influence of SRA programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. *International Journal of Technology and Design Education, 31*(2), 203–222. https://doi.org/10.1007/s10798-019-09559-9

Garaika, G., Margahana, H. M., & Negara, S. T. (2019). Self efficacy, self personality and self confidence on entrepreneurial intention: Study on young enterprises. *Journal of Entrepreneurship Education, 22*(1), 1–12.

Gomes, A., & Mendes, A. (2014, October). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings* (pp. 1–8). IEEE. https://doi.org/10.1109/FIE.2014.7044086

Hair, J. F., Black, W. C., Babin, B. J., Anderson, R. E., & Tatham, R. L. (2006). Multivariate data analysis 6th Edition. *Pearson Prentice Hall. New Jersey. humans: Critique and reformulation. Journal of Abnormal Psychology*, *87*, 49–74.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175–192. https://doi.org/10.1007/s10956-016-9663-z

Kalita, G. (2021). Analyzing the Level of Self Confidence of the Post Graduate Students in Relation to Certain Variables. *Psychology and Education Journal*, *58*(3), 1381–1383. https://doi.org/10.17762/pae.v58i3.3870

Kittur, J. (2020). Measuring the programming self-efficacy of Electrical and Electronics Engineering students. *IEEE Transactions on Education, 63*(3), 216–223. https://doi.org/10.1109/TE.2020.2975342

Kukul, V., Gökçearslan, Ş., & Günbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama*, *7*(1), 158–179. https://doi.org/10.17943/etku.288493

Liu, C., He, J., Ding, C., Fan, X., Hwang, G. J., & Zhang, Y. (2021). Self-oriented learning perfectionism and English learning burnout among EFL learners using mobile applications: The mediating roles of English learning anxiety and grit. *Learning and individual Differences*, *88*, 102011. https://doi.org/10.1016/j.lindif.2021.102011

Ozuorcun, N. C., & Tabak, F. (2012). Is m-learning versus e-learning or are they supporting each other? *Procedia-Social and Behavioral Sciences, 46*, 299–305. https://doi.org/10.1016/j.sbspro.2012.05.110

Peel, A., Sadler, T. D., & Friedrichsen, P. (2022). Algorithmic Explanations: an Unplugged Instructional Approach to Integrate Science and Computational Thinking. *Journal of Science Education and Technology*, 1–14. https://doi.org/10.1007/s10956-022-09965-0

Rich, P. J., Larsen, R. A., & Mason, S. L. (2021). Measuring teacher beliefs about coding and computational thinking. *Journal of Research on Technology in Education, 53*(3), 296–316. https://doi.org/10.1080/15391523.2020.1771232

Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K-12: In-service teacher perceptions of computational thinking. In *Computational thinking in the STEM disciplines* (pp. 151–164). Springer, Cham. https://doi.org/10.1007/978-3-319-93566-9_8

Schwarzer, R., & Warner, L. M. (2013). Perceived self-efficacy and its relationship to resilience. In *Resilience in children, adolescents, and adults* (pp. 139–150). Springer, New York, NY. https://doi.org/10.1007/978-1-4614-4939-3_10

Shanmugam, L., Yassin, S. F., & Khalid, F. (2019). Enhancing students' motivation to learn computational thinking through mobile application development module (M-CT). *International Journal of Engineering and Advanced Technology, 8*(5), 1293–1303.

Standl, B., & Schlomske-Bodenstein, N. (2021, November). Exploring Indicators to Promote Preservice Teachers' Self-Efficacy in Programming Tasks. In *21st Koli Calling International Conference on Computing Education Research* (pp. 1–3). https://doi.org/10.1145/3488042.3489965

Tsai, C. Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior, 95*, 224–232. https://doi.org/10.1016/j.chb.2018.11.038

Wu, L., Looi, C. -K., Liu, L., & How, M. -L. (2018). Understanding and developing in-service teachers' perceptions towards teaching in computational thinking: Two studies. In *Proceedings of the 26th International Conference on Computers in Education* (pp.735–742). Philippines: Asia-Pacific Society for Computers in Education.

Yilmaz Ince, E., & Koc, M. (2021). The consequences of robotics programming education on computational thinking skills: An intervention of the Young Engineer's Workshop (YEW). *Computer Applications in Engineering Education, 29*(1), 191–208. https://doi.org/10.1002/cae.22321

Zhang, J. H., Meng, B., Zou, L. C., Zhu, Y., & Hwang, G. J. (2021). Progressive flowchart development scaffolding to improve university students' computational thinking and programming self-efficacy. *Interactive Learning Environments*, 1–18. https://doi.org/10.1080/10494820.2021.1943687

Zwart, D. P., Noroozi, O., Van Luit, J. E., Goei, S. L., & Nieuwenhuis, A. (2020). Effects of Digital Learning Materials on nursing students' mathematics learning, self-efficacy, and task value in vocational education. *Nurse Education in Practice*, *44*, 102755. https://doi.org/10.1016/j.nepr.2020.102755

Authors and Affiliations

Attila Kovari[1,2,3] · Jozsef Katona[4,5]

[1] Institute of Digital Technology, Faculty of Computer Science, Eszterházy Károly Catholic University, Eger, Hungary

[2] Institute of Electronics and Communication Systems, Kandó Kálmán Faculty of Electrical Engineering, Obuda University, Budapest, Hungary

[3] GAMF Faculty of Engineering and Computer Science, John Von Neumann University, Kecskemét, Hungary

[4] Department of Software Development and Application, Institute of Computer Engineering, University of Dunaujvaros, Dunaujvaros, Hungary

[5] Trefort Ágoston Center for Engineering Education, Institute of Electrophysics, Kandó Kálmán Faculty of Electrical Engineering, Obuda University, Budapest, Hungary