International
Journal of Human Sciences
ISSN:2458-9489

Volume 14 Issue 4 Year: 2017

Could rapid application development tools be used as icebreakers in programming education?

Mutlu Tahsin Üstündağ[1]

Abstract
Today, learning how to program or coding is an important issue even for children. So, pre-service Information Technology (IT) teachers are expected to have gained necessary skills for teaching programming. Considering that learning programming is not an easy process, we need icebreakers in order to change pre-service teachers' perceptions of programming positively. This study focused on an implementation of a training of rapid application development (RAD). The aim of the study was to come up with an answer to the question; "can RAD be used as an icebreaker in order to change Turkish pre-service IT teachers' perceptions towards programming positively?" The findings revealed that RAD tools can be used as icebreakers in the context of the study. In this respect, it is recommended that these tools be included in the higher education programs providing informatics education.

Keywords: Rapid application development; programming; teaching; self-efficacy; icebreaker.

1. Introduction

In the 21th century, countries started to teach even children technology and programming. Rapid changes in computer science and the need for productive people made it necessary to teach children at least fundamentals of programming. Importance of teaching programming in early years was already mentioned in the literature (Kafai & Burke, 2013). Also, curriculums were updated in many countries considering this issue (Kalelioğlu, Gülbahar, 2014; Kalelioğlu, 2015; Lee, Martin & Apone, 2014; Sáez-López, Román-González & Vázquez-Cano, 2016). Some tools such as Scratch and App Inventor are being used in order to teach how to program and they seem to be successful in terms of learning outcomes (Gouws, Bradshaw & Wentworth, 2013; Goadrich, 2014). For this reason, it must be taken into consideration that we need teachers having sufficient competencies of teaching programming in order to be successful in educating children in this manner.

When it comes to Turkey context, Turkish Ministry of Education defined competencies of information technology teachers (IT teachers) two of which are being able to adapt to technologies by knowing the effects of new technologies in the society and designing and using technology-supported learning environments that can meet the different needs of the learners. In schools, IT teachers are expected to teach fundamentals of programming. At this point, pre-service IT teachers are expected to have gained necessary skills for teaching programming. On the other hand, teaching programming seems to be a controversial issue in the literature because it is not an easy process to cope with and students experience difficulties (Anastasiadou & Karakos, 2011; Baser, 2013; Du Boulay, 1986; Milne & Rowe, 2002; Mow, 2008) and is related to many factors such as self-efficacy (Ramalingam & Wiedenbeck, 1998; Yükselturk & Altıok, 2017), attitude (Baser, 2013) and

[1] Ph.D., Gazi University, Computer Education and Instructional Technologies, mutlutahsin@gmail.com

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

perception of programming (Zainal, Shahrani, Yatim, Rahman, Rahmat & Latih, 2012) etc. Considering the previous research on teaching programming, it can be argued that students usually have difficulty in learning. Students with low self-efficacy of programming tend to perceive problems that are more difficult than their tasks and therefore cannot solve problems (Aşkar & Davenport, 2009). Negative self-efficacy perceptions result in difficulty and failure. For this reason, it is important for learners to have positive perceptions to be successful in learning programming (Altun & Mazman, 2012; Baser, 2013). Experienced difficulties have the potential to become barriers for learning programming (Özoran, Çağıltay & Topallı, 2012). According to the above-mentioned issues, although teaching programming is a complex issue, literature still lacks research findings regarding easy and practical ways of teaching programming or eliminating negative perceptions of programming. At this point, rapid application development tools, which may affect attitudes, self-efficacy and perceptions in a positive manner, come into prominence.

Rapid Application Development (RAD) is among the various models that have been developed to make programming or software development processes more efficient (Munassar & Govardhan, 2010). RAD, one of the software development models, was conceptually used for the first time by James Martin in his book. He described RAD as a software development cycle that can develop software faster and better than conventional methods (Martin, 1991). RAD is a method that makes the analysis, design, building and testing phases efficient with short and repetitive development cycles (Hashim & Mohamed, 2013). Advantages of RAD can be listed as; ease of application, superior user satisfaction and moving to the market in a shorter time (Daud, Bakar & Rusli, 2010).

Today, there are many RAD tools that can be used to develop quick applications to both business environments and teaching environments. The type of software to be used varies according to the purpose. For example, Oracle APEX, Spring Roo, RAD Studio XE6 and Visual Studio LightSwitch are some of the RAD tools that can be used to develop rapid web applications. Visual Studio LightSwitch makes it easy to create data-centric business applications that can work with many data sources and create clients to work across a variety of tools. Writing, building and developing a simple web application with pages that only create, read and update in a database is a time consuming and expensive process. Leung (2015) states that using LightSwitch enables the programmer to develop applications quickly and easily.

When learners have negative experiences during learning algorithms and programming in a traditional way, their attitudes and achievements are affected negatively. On the other hand, motivation and attitudes are critical factors for learners of programming (Anastasiadou & Karakos, 2011; Erol & Kurt, 2017). Morrison and Newman (2001) stated that prior programming experience is an important issue to be considered because positive experience have positive impact on students' achievement in programming. In addition, attitudes of students towards programming are still a contemporary issue because of its effects on learning programming. For example, Cetin and Ozden (2015) developed a computer programming attitude scale for university students. Also, the authors recommended further research on this topic. According to Gençtürk and Korucu (2017), Turkish pre-service IT teachers from Computer Education and Instructional Technologies (CEIT) undergraduate program do not have adequate experience and knowledge regarding programming. Their study aimed to investigate the effects of utilizing web 2.0 technologies on the success and attitudes of CEIT students towards programming. Findings of this study revealed that this intervention made positive effects on success and attitudes. In another study on Turkish pre-service IT teachers conducted by Yükseltürk and Altıok (2017), the Scratch tool was utilized in order to eliminate negative attitudes of pre-service IT teachers towards programming and their self-efficacy levels. The implementation was presented to be successful in this manner. There are many other research dealing with difficulties or challenges in leaning programming (Shaw, 2013; Lau & Yuen, 2011; Yurdugül & Aşkar, 2013; Jegede, 2009; Aşkar & Davenport, 2009).

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

As mentioned above, there is a robust literature on difficulties or challenges of teaching programming and many factors like self-efficacy, perception or attitude closely related to this issue. However, there is little amount of research regarding utilizing RAD tools in programming education in this manner although RAD has the potential to be successful in developing software applications in terms of many aspects, as mentioned above. Some examples can be given regarding the use of RAD in educational settings:

Instructional designers started to use RAD for developing instructional materials. RAD contains development processes that enable a primitive prototype to be transformed into a fully developed product (Lohr, Javeri, Mahoney, Gall, Li & Strongin, 2003). Lohr et al., (2003) focused on the effects of utilizing RAD processes on usability of instructional materials developed by pre-service teachers. Findings of the study presented positive effects. Furthermore, they recommend further research regarding the use of RAD tools in educational settings.

Lee (2011), through visual programming software called Etoys, has made a study to ensure that teachers can develop application software to serve teaching purposes. According to the findings, attitudes of the teachers towards the implementation were found to be positive. In addition, participants indicated that they would like to continue to use the application. Huaqing and Li (2011) have developed a system called "Rapid Software Development Platform" to make the programming process more regular and faster, and have found that the platform they developed as a result of their work is more effective than the normal software development process. Daud et al. (2010) developed a system using the RAD method, in which students can upload their work and receive feedback from the teacher. This system could be rearranged according to the feedback from teachers and students.

In brief, RAD can be utilized in order to cope with negativity regarding student perception, self-efficacy or attitude towards programming. In other words, RAD can be used as an icebreaker in this manner. In this study, "icebreaker" is used for tool which have the potential to decrease the negative attitudes and perceptions and to increase the level of perceived self-efficacy towards programming. RAD tools enable easy and fast development of software as mentioned above. This potential can be taken as an opportunity for encouraging positive attitudes or perceptions towards programming. This research focuses on pre-service IT teachers' perceptions of programming. This is accepted as an important issue, because today's pre-service IT teachers will be teaching programming or coding to students of 21th century in the near future. For this reason, we must search for strategies to ensure that these teachers have positive attitudes and perceptions of programming so that they will be successful in teaching programming. The study investigates pre-service IT teachers' self-efficacy levels of educational software development and programming and then implements a RAD training on LightSwitch tool in order to examine its effects. For this reason, the study was conducted in three stages in order to investigate (1) educational software development self-efficacy levels of IT teachers, (2) programming self-efficacy and knowledge levels of them and (3) effects of the RAD (LightSwitch) training.

2. Purpose

The study intends to come up with an answer to the question; can RAD be used as an icebreaker in order to change Turkish pre-service IT teachers' perceptions towards programming positively? For this main problem, the research investigated;

- 196 pre-service IT teachers' (from different universities) self-efficacy levels of educational software development,
- 35 pre-service IT teachers' (from Gazi University) self-efficacy levels of programming,
- 35 pre-service IT teachers' (from Gazi University) knowledge levels of programming,
- the change in the perceptions of 19 pre-service IT teachers (from Gazi University) towards programming after completing a LightSwitch RAD training,

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

- views and opinions of 19 pre-service IT teachers, who completed the training, regarding RAD tools.

3. Method

This research was carried out with mixed method design in which qualitative and quantitative methods were used together. This method provides the deeper investigation of the research problem (Cresswell, 2008). Cresswell (2008) approached mixed design in four categories namely embedded, explanatory, exploratory and parallel. This research was implemented on pre-service IT teachers in three stages in 2015-2016 academic year, spring semester. Each stage was actualized with different data sets and analysis methods within itself. The qualitative and quantitative data collection tools were applied in different times in each stage during the implementation process. After the quantitative data were collected and analyzed, the qualitative data was analyzed. For this reason, the study was conducted as an "explanatory mixed method" research. Detailed information regarding the stages of the research is presented under implementation title:

3.1. Implementation

The three stages of the research which was implemented in 2015-2016 academic year, spring semester are presented in Figure 1:

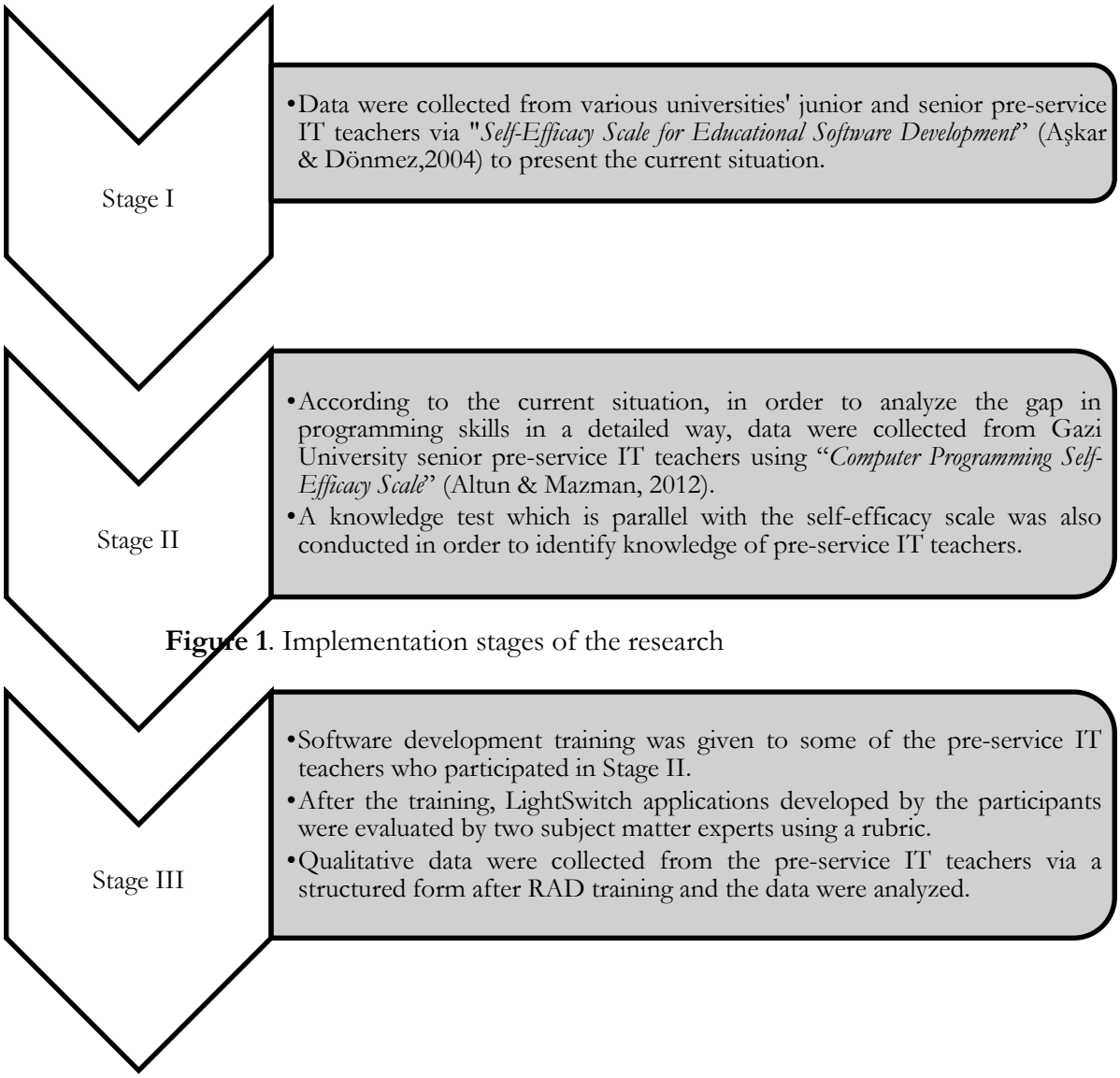


Figure 1. Implementation stages of the research

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Figure 1 represents the three stages of the implementation process. The aim of Stage I was to investigate the current status in terms of pre-service IT teachers' perceived educational software development self-efficacy levels. Stage II intended to identify the level of pre-service IT teachers' perceived computer programming self-efficacy. According to the current situation, in order to analyze the gap in programming skills in a detailed way, data were collected from Gazi University senior pre-service IT teachers using "Computer Programming Self-Efficacy Scale" (Altun & Mazman, 2012). Furthermore, a knowledge test which is parallel with the self-efficacy scale was also conducted in order to identify the knowledge of pre-service IT teachers. These two stages helped the researcher point out the gap regarding these issues. This was quite important, because the above mentioned problem statement of the research focused on utilizing RAD as an icebreaker in this manner, if there is really a need for changing pre-service IT teachers' perceptions towards programming positively. Finally, Stage III included the RAD training and analysis of the data derived from the participants of the training in order to clarify the results of the training implementation. After the training, LightSwitch applications developed by the participants were evaluated by two subject matter experts using a rubric.

3.2. Participants

Participants of the study are presented according to the stages because the study was conducted in three stages.

3.2.1. Participants of Stage I

The university and grade levels of junior and senior pre-service IT teachers who participated in Self-Efficacy Scale for Educational Software Development in the first stage of the implementation are given in Table 1.

Table 1. The descriptive statistics of participants of self-efficacy scale for educational software development

University	Grade Junior		Senior		Total	
	N	%	N	%	N	%
Abant İzzet Baysal University	6	3.06	1	0.51	7	3.57
Ahi Evran University	39	19.90	29	14.80	68	34.69
Anadolu University	1	0.51	1	0.51	2	1.02
Gazi University	23	11.73	37	18.88	60	30.61
Hacettepe University	1	0.51	0	0.00	1	0.51
Kırıkkale University	15	7.65	0	0.00	15	7.65
Necmettin Erbakan University	22	11.22	13	6.63	35	17.86
Uludağ University	1	0.51	7	3.57	8	4.08
Total	108	55.10	88	44.90	196	100

When the table is examined, it can be seen that the highest rate of participants are from Ahi Evran University (34.69%) followed by Gazi University (30.61%) and Necmettin Erbakan University (17.86%). The least participation rate is from Hacettepe University (0.51%). The scale was conducted on a total number of 196 pre-service IT teachers from 8 universities. According to Table 1, it is obvious that participation from some of the universities seem to be low as the scale was filled in by the participants on a voluntary basis. This can be assumed as a limitation of the study.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

3.2.2. Participants of Stage II

In the second stage of the implementation senior pre-service IT teachers from Gazi University participated in the research and the data were collected via the *Computer Programming Self-Efficacy Scale* and *the Programming Knowledge Test*. Demographic information about the participants is presented in Table 2:

Table 2. Descriptive statistics of participants of Computer Programming Self-Efficacy Scale and the Programming Knowledge Test

Gender	N	%
Female	16	45.70
Male	19	54.30
Total	35	100

As seen from Table 2, the number of male participants (19) is higher than female participants (16).

3.2.3. Participants of Stage III

Totally 19 Gazi University senior pre-service IT teachers, attending both the first and the second stage of the implementation, were given an application development training using RAD tools. Number of the participants of Stage 2 is lower than Stage 1 due to the limitations regarding the training, during 6 weeks, which includes deep participation, developing software and other activities taking too much time and requiring hardware opportunities. In addition, after the RAD training, qualitative data via structured forms were collected from the participants. Demographic information about the participants of the third stage is given in Table 3:

Table 3. Descriptive statistics of RAD training participants

Gender	N	%
Female	11	57.89
Male	8	42.11
Total	19	100

Participants of the RAD training that can be seen in Table 3 are 57.89% female and 42.11% male with a total of 19 participants.

3.3. Data Collection Tools

Data collection tools of the study are presented in sub-titles because the study was conducted in three stages.

3.3.1. Stage I

The data collection tool used in the first stage was *Self-Efficacy Scale for Educational Software Development* which was developed by Aşkar and Dönmez (2004). In order for determining the scale items, the researchers analyzed the different Educational Software Development processes and made some interviews within some institutions in which software were developed. At the end of this analysis, 6 dimensions were determined to be taken into account in developing educational software (project management, instructional design, graphic design, animation design, programming and sound-video design) and items about them were written. The prepared 22 items were applied to junior students and after the analyses; no change was made in the items. Students responded to the situations in the items ranging from "strongly trust" to "strongly distrust" according to 100 point numerical rating scale. The final form of the scale was administered to 283 junior and senior students. For determining the validity of the scale, principal components factor analysis with

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

varimax rotation was used. According to the analysis results, four factors were determined as; "project management and instructional design", "animation and sound-video design", "graphics design" and "Programming". The reliability coefficient of the scale was calculated with Cronbach Alpha and was found as .92.

3.3.2. Stage II

The data collection tool used in the second stage, *Computer Programming Self-Efficacy Scale*, was developed by Ramalingam and Wiedenbeck in 1998 in order to evaluate the perceived self-efficacy of university students in computer programming. The scale was adapted into Turkish by Altun and Mazman in 2012. The original scale by Ramalingam and Wiedenbeck was composed of 32 items prepared in 7 Likert type. According to the exploratory factor analysis, the scale was composed of four items; "independence and persistence", "ability to perform simple programming tasks", "ability to perform complex programming tasks" and finally "self-regulation" (Mazman and Altun, 2013; Altun and Mazman, 2012). However, in the adapted version of the scale into Turkish by Altun and Mazman, the scale consisted of 9 items with 2 factors which are; "ability to perform simple programming tasks" and "ability to perform complex programming tasks" in 7 Likert type. Simple programming tasks factor consisted of 3 items (such as basic level mean computing, message writing on the screen) whereas complex programming tasks factor consisted of 6 items (such as error debugging, working on multiple files, rewriting an algorithm). Cronbach alpha coefficient of the scale was found as .928.

In addition to programming self-efficacy, the knowledge test which is parallel with the self-efficacy scale was also conducted in order to identify programming knowledge of pre-service IT teachers. This knowledge test included 6 questions. The knowledge test was developed by the researcher and revised according to two subject matter experts' views. In order to collect data from the experts, an expert evaluation form was utilized. This form included questions for evaluating the knowledge test in terms of content, quality and applicability. 3 questions of the knowledge test were related to ability to perform simple programming tasks. For example; "Write the codes of the program that gives the average of the 3 numbers entered from the keyboard". Other 3 questions were related to ability to perform complex programming tasks. For example; "You are expected to write the program of the lift system of a building with 10 floors. The working principle of elevators is as follows: 2 lifts can be called for each floor with a single control center. The most basic principle is to ensure that the closest elevator comes to the requested floor. If the elevators have equal distances, the direction of movement of the person is considered."

3.3.3. Stage III

After the training, LightSwitch applications developed by the participants were evaluated by two subject matter experts using a rubric. This rubric was developed by the researcher according to the subject matter experts' views. The evaluation was carried out with two experts and the average score of the experts was taken.

In this stage of the research qualitative data collection tool developed by the researcher and confirmed by the experts were used. Pre-service IT teachers' views were gathered with a structured form in which there were a total of 10 questions consisting of 8 multiple-choice and 2 open-ended questions.

3.4. Data Analysis

The quantitative data were analyzed with IBM SPSS 21.0 and Microsoft Excel 2016 programs. Whether the mean scores of dependent groups are significant or not was tested in 0.05 significance level and .95 probability confidence interval was obtained. Independent sample t test was used for analyzing the scores the pre-service teachers got from Self-Efficacy Scale for Educational Software Development in terms of grade level. Scores pre-service teachers taken from the Computer Programming Self-Efficacy Scale were analyzed using descriptive statistics. After the

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

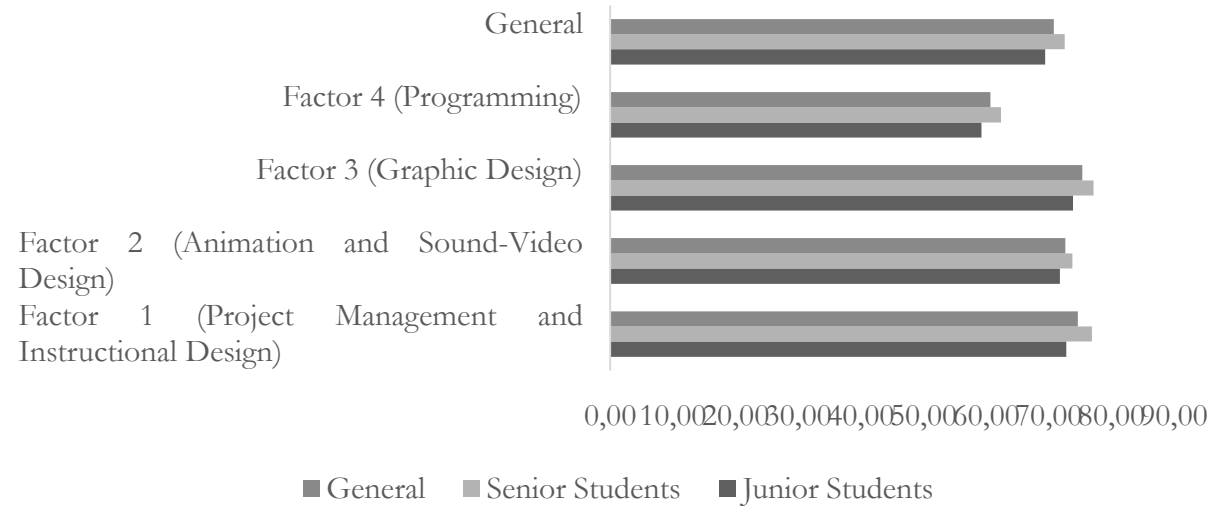
evaluation for the programming knowledge test, the analysis was done using the descriptive analysis. The scores obtained after the product evaluation were again analyzed using descriptive analyzes. The qualitative data was investigated using content analysis method. In order to provide the coder reliability, the codes were analyzed by two experts independently. The determined codes were compared and dissensus situations were discussed. For coder reliability, the reliability formula determined by Miles and Huberman (1994) was used. The situation was tried to be summarized as it is by giving direct quotations from pre-service teachers' answers (Yıldırım & Şimşek, 2006). Instead of using pre-service teachers' names codes such as P1 and P2 were used in the direct quotations.

4. Findings

Findings of the study are presented in sub-titles because the study was conducted in three stages.

4.1. Stage I

In order to clearly present the current situation, data from various universities' junior and senior pre-service IT teachers were collected via *Self-Efficacy Scale for Educational Software Development*. The collected data were presented in terms of general and sub-factors in Graphic 1:



Graphic 1. The presentation of the scores acquired via self-efficacy scale for educational software development

When Graphic 1 is examined, the overall average of all factors for all participants is 74.66, 72.68, 75.37 and 60.68, respectively. The average scale score is 70.85. It is seen from Graphic 1 that Factor 4 (Programming) is considerably lower than the other dimensions. In addition to this, another subject of curiosity is whether there is a meaningful difference between the grade levels. Therefore, t-test results for independent samples are presented in Table 4:

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Table 4. T-Test results of the scores of pre-service IT teachers' responses to self-efficacy scale for educational software development in terms of grade level

Factors	Measurement	N	X	Sd	df	t	p
Factor 1 (Project Management and Instructional Design)	Junior	108	72.82	14.33	194	2.014	.045
	Senior	88	76.91	13.89			
Factor 2 (Animation and Sound-Video Design)	Junior	108	71.80	18.57	194	.752	.453
	Senior	88	73.77	17.80			
Factor 3 (Graphic Design)	Junior	108	73.90	17.64	194	1.262	.209
	Senior	88	77.16	18.44			
Factor 4 (Programming)	Junior	108	59.43	22.80	194	.956	.340
	Senior	88	62.48	21.37			
General	Junior	108	69.49	14.79	194	1.402	.163
	Senior	88	72.58	16.03			

When table 4 is examined it is seen that no difference was seen in all factors except from Factor 1 (Project Management and Instructional Design) and the general scores in terms of grade level. The fact that project management courses are presented in senior grade in CEIT curricula could be the reason for this difference. Stage I was related to 196 pre-service IT teachers' (from different universities) self-efficacy levels of educational software development. Findings showed that Factor 4 (Programming) factor average score was the lowest one. For this reason, Stage II focused on programming self-efficacy.

4.2. Stage II

The current situation that was put forward in the first stage showed that a self-efficacy perception of the pre-service IT teachers towards educational software development was low. This finding seems to be compatible with previous research (Gençtürk & Korucu, 2017; Aşkar & Davenport, 2009). In this stage of the research, *Computer Programming Self-Efficacy Scale* was applied to pre-service teachers.

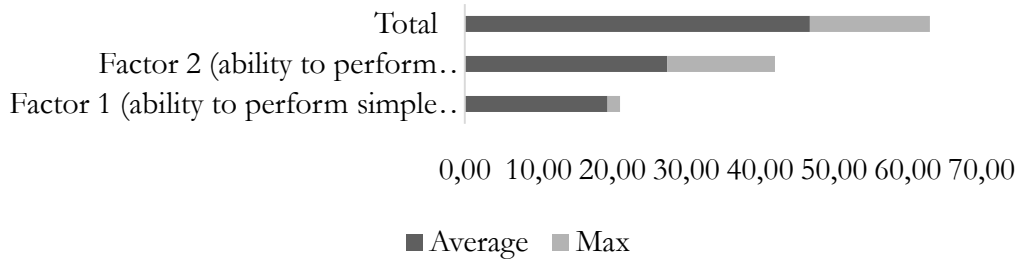
In order to analyze the scores given to the scale is presented as factors and as a total in Table 5:

Table 5. Descriptive statistics of scores of computer programming self-efficacy scale

Factors	N	Max.	X	sd
Factor 1 (ability to perform simple programming tasks)	35	21.00	19.31	3.60
Factor 2 (ability to perform complex programming tasks)	35	42.00	27.40	7.37
Total	35	63.00	46.71	9.47

When table 5 is examined it is seen that self-efficacy of pre-service IT teachers Factor 1 (ability to perform simple programming tasks) is quite high (19.31 /21.00) whereas the result is different for Factor 2 (ability to perform complex programming tasks).
In order to show the difference clearly, Graphic 2 is presented:

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155



Graphic 2. Descriptive statistics of the scores of computer programming self-efficacy scale

It is clearly seen when Graphic 2 is examined that computer-programming self-efficacy of pre-service IT teachers in performing complex programming tasks is low (27.40 out of 42). Findings related to pre-service IT teachers programming knowledge test are presented in Table 6:

Table 6. Descriptive statistics of scores of computer programming knowledge test

Factors	N	X	sd
Factor 1 (ability to perform simple programming tasks)	35	94.43	7.65
Factor 2 (ability to perform complex programming tasks)	35	37.71	19.19
Total	35	66.07	13.06

Table 6 shows that factor 1 (ability to perform simple programming tasks) is similar to computer programming self-efficacy scale. Pre-service IT teachers seem to be successful at this factor (X = 94.43) However, the success rate for factor 2 (ability to perform complex programming tasks) was found to be very low (X = 37.71). Considering the findings for Stage II, Stage III focused on the change in the perceptions of 19 pre-service IT teachers (from Gazi University) towards programming after completing a LightSwitch RAD training and views and opinions of them regarding RAD tools.

4.3. Stage III

Findings for this stage were reached by analyzing the data derived from the participants of the LightSwitch training. The applications they developed using LightSwitch were evaluated via the rubric and the findings are presented in Table 7:

Table 7. Descriptive statistics of scores derived by the rubric

Measurement	N	X	sd
Score	19	88.82	7.70

Table 7 shows that pre-service IT teachers have achieved a very high success (X=88.82). All of the pre-service IT teachers completed a product successfully. It has been observed that 12 participants almost completed this process without any mistakes.

Pre-Service teacher views on the question "(please) indicate your level of satisfaction in LightSwitch training" are given below.

Percentages and frequencies of pre-service IT teachers' views on the satisfaction levels about the training they attended were consulted are presented in Table 8.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Table 8. Satisfaction levels of pre-service IT teachers about the training they attended

	f	**%**
Not satisfied at all	0	0
Dissatisfied	0	0
Neutral	0	0
Satisfied	3	15.8
Very satisfied	16	84.2
Total	19	100

When the table about the satisfaction levels of the pre-service IT teachers is examined, it is seen that the satisfaction levels are at a high level as 16 of them stated that they were "very satisfied" and 3 stated that they were "satisfied".

Pre-service IT teachers' views on the question "Do you believe that you can now develop faster and more effective training software when compared to your previous situation without attending the training?":

Previous research focusing on interventions using various tools for changing views or attitudes of learners also indicate that well-designed trainings seem to be successful in this manner (Gençtürk & Korucu, 2017; Yükseltürk & Altıok, 2017).

Percentages and frequencies of pre-service IT teachers' views on the question whether they believe that they can now develop faster and more effective training software when compared to their previous situation without attending the training are presented in Table 9.

Table 9. Whether the pre-service IT teachers believe if they can now develop faster and more effective training software after attending the program

	f	**%**
Definitely no	0	0
No	0	0
Neutral	0	0
Yes	5	26.3
Definitely yes	14	73.7
Total	19	100

When the table about the beliefs of pre-service teachers whether they can now develop faster and more effective software after attending the training program shows that their belief in themselves towards developing faster and more effective software using this program is high, as 14 of them stated "definitely yes" and 5 stated "yes".

Pre-Service teacher views on the question "learning LightSwitch is …":

Percentages and frequencies of pre-service IT teachers' answers to the question "learning LightSwitch is …" is presented in Table 10.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Table 10. Answers to the question "learning LightSwitch is …"

	f	**%**
Very easy	5	26.3
Easy	9	47.4
Neutral	2	10.5
Difficult	2	10.5
Very difficult	1	5.3
Total	19	100

It is seen from the table that pre-service IT teachers' responses to the question "Learning LightSwitch…" are as follows: 5 of them stated "very easy", 9 reported "easy", 2 were neutral, 2 stated as "difficult" and 1 pre-service teacher reported as "very difficult". This shows that a majority of the pre-service teachers did not have any difficulty in learning this program, on the contrary, they could better be adapted and learnt compared to other programming skills.

Pre-Service teacher views on the question "what have you gained with this training?"

Values the pre-service teachers stated to the question what this training attained them are given in Table 11.

Table 11. Themes and codes about what the pre-service teachers have gained with this training

Theme	Code	f
Cognitive	Provided me in developing software fast	6
	I designed what I did in past as fragmentary now as a whole	5
	It provided too much complex application development with less code knowledge	7
	Lead up the development of new and different software development	2
Affective	I saw that we had a broader perspective	3
	I developed software without being stressed and afraid but pleasingly	5
	Increased my confidence to do programming	7
	I was motivated when I saw the products	2

As can be seen from the table, when qualitative data obtained from the research is grouped according to the similar qualities, two themes namely cognitive and affective have arisen. These are:

1. Views in cognitive level

Pre-service IT teachers taking the course emphasized in terms of cognitive level that; provided in developing software fast, helped them to design as a whole what they did fragmentary in the past, provided too much complex application development with less code knowledge and lead up the development of new and different software development.

a. In terms of the code: "Provided me in developing software fast (n=6)" pre-service teachers stated views defining the role of LightSwitch software as providing them with fast, in a shorter time and as a whole contribution. For example a pre-service teacher stated as:

P1: "…Provided me to construct a system fast and easily…" whereas another pre-service teacher evaluated it as:

P9: "I realized that I could develop a software in a very short time and a little effort…"

b. In terms of the code: "I designed what I did in past as fragmentary now as a whole (n=5)" pre-service teachers stated views that LightSwitch provided them with a vision on how to develop a system at all points. For instance a pre-service teacher stated his view as:

P3: "…I did as a project what I did in past as fragmentary." Whereas another pre-service teacher explained his views as:

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

P19: "Lead up the development of a project with all its aspects".

c. In terms of the code: "It provided too much complex application development with less code knowledge (n=7)" pre-service teachers reported that with LightSwitch program, it is possible to make advanced implementations with basic coding knowledge.

P12 stated his views as: "… I liked that I could develop a software without coding knowledge" whereas another pre-service teacher stated:

P18: "…I learnt that I could also develop a software by writing fewer codes."

d. In terms of the code: "Lead up the development of new and different software development (n=2)" pre-service teachers stated their views on what could be done in the future with LightSwitch program. For instance a pre-service teacher reported his views as:

P6: "…developing rapid software directed me to develop more software…"

2. Views in Affective Level

a. In terms of the code: "I saw that we had a broader perspective (n=3)" pre-service teachers reported the effect of LightSwitch program in attaining different points of view. For example; a view of a pre-service teacher was:

P18: "… I saw that I could develop a software without writing codes, which opened up my horizon…" whereas another view was:

P17: "… I learnt by living what could be done with lesser code knowledge."

b. In terms of the code: "I developed software without being stressed and afraid but pleasingly (n=5)" pre-service teachers stated that with LightSwitch program they could develop a software without any fear but fondly and willing fully. For example a pre-service teacher explained her views as:

P2: "It made us learn software development without being stressed" and another pre-service teacher stated his views as:

P11: "I liked to be able to develop software without coding knowledge".

c. In terms of the code: "Increased my confidence to do programming (n=7)" pre-service teachers reported that with LightSwitch they felt the feeling of doing something more self-confidently. Views on this respect are as follows:

P14: "I saw that a software could be developed in a very short time and less effort. Therefore my self-confidence in my field increased."

P19: "First of everything, this program provided us to rely on ourselves."

d. In terms of the code: "I was motivated when I saw the products (n=2)" pre-service teachers told that with the products that came out of LightSwitch they were motivated to develop much software. A view of a pre-service teacher is as follows:

P19: "… that we have learnt it in a very short time and a very nice concrete product came out motivated us…"

Pre-Service teacher views on the question "Do you think that all CEIT graduates should have the knowledge of a rapid software development?"

The responses of the pre-service teachers whose views were consulted to this question is presented in Table 12:

Table 12. Pre-service IT teachers' views on whether all CEIT graduates should have the knowledge of rapid software development

	f	%
Definitely no	0	0
No	0	0
Neutral	0	0
Yes	1	5.3
Definitely yes	18	94.7
Total	19	100

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

When the table is examined it is seen that 18 pre-service teachers replied as "definitely yes" and 1 pre-service teacher responded with "yes". This shows that pre-service teachers stated a predominant opinion for a CEIT grade to have at least one rapid software development.

Pre-Service teacher views on the question "Do you think there should be course/s on rapid software development?"

The responses of the pre-service teachers to the question whether there should be course/s on rapid software development is presented in Table 13:

Table 13. Views of pre-service IT teachers on whether there should be course/s on rapid software development

	f	%
Yes	18	94.7
Neutral	0	0
No	1	5.3
Total	19	100

When the frequencies of the pre-service teachers' responses about whether there should be course/s on rapid software development, it can be seen that the majority of the pre-service teachers (18) responded as "yes" so, there should be course/s and only one answered "no".

Pre-service teacher views on the question "Did the 4 year training you had in CEIT department make you gain the skill of developing software that could bring a solution to real life problems?"

The responses of the pre-service teachers whose views were consulted to the question whether the 4 year program acquired them with skill of developing soft wares that could bring a solution to real life problems are presented in Table 14:

Table 14. Views of pre-service IT teachers about their CEIT training's effect on providing them with the skill of developing software that could bring a solution to real life problems

	f	%
Definitely no	0	0
No	2	10.6
Neutral	6	31.8
Yes	8	41.7
Definitely yes	3	15.9
Total	19	100

When the table is examined, 3 pre-service teachers responded as "definitely yes"; 8 as "yes", 6 of them answered as "neutral" and 2 pre-service teachers responded as "no". So, some of the pre-service teachers stated positive views and some stated negative views whereas some were neutral on the subject.

Pre-Service teacher views on the question "How much were you confident in yourself in programming (application development) before attending LightSwitch training?

The responses of pre-service teachers about their confidence situations before attending the LightSwitch training are presented in Table 15:

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Table 15. Views of pre-service IT teachers about their confidence situations before attending the LightSwitch training

	f	%
I was not confident at all	0	0
I was not confident	8	41.7
Neutral	7	37.1
I was confident	3	15.9
I was very confident	1	5.3
Total	19	100

When the table is examined, it can be seen that some felt themselves inadequate, some were neutral and a very few felt themselves competent. The frequencies are as follows; 8 pre-service teachers "I was not confident at all", 7 "neutral", 3 of them "I was confident" and 1 "I was very confident".

Pre-Service teacher views on the question "How much do you trust yourself in programming (application development) after you had taken LightSwitch training?"

The views of the pre-service teachers on how much they feel confidence in themselves after taking the LightSwitch training is presented in Table 16.

Table 16. Views of pre-service IT teachers on the effect of LightSwitch training to their self-confidence

	f	%
I am not confident at all	0	0
I am not confident	0	0
Neutral	0	0
I am confident	11	58.3
I am very confident	8	41.7
Total	19	100

When the frequencies of the responses of pre-service teachers to the question "how much they are confident in themselves after attending the LightSwitch training" is examined; it can be seen that 8 pre-service teachers responded as "I am very confident" and 11 of them responded as "I am confident" which shows that the whole class feel themselves competent after attending this training.

Pre-Service teacher views on the question "Please indicate if you have view, suggestion or criticisms on this topic"

The responses of the pre-service teachers about their further view, suggestions and criticisms are given in Table 17.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Table 17. Further view, suggestion and criticisms of pre-service teachers

Code	f	Quotations from sample pre-service teacher views
A practical program, should be taught when basic programming skills are attained before senior grade	7	P5: "…But I am upset to have taken this training in senior grade. I wish I had learnt in the earlier grades and worked on better things." P12: "Students need to be introduced LightSwitch after attaining a certain programming basis."
Should be taught as a compulsory course	4	P14: "This training should be a compulsory course in CEIT. After all, there are many areas of interest in software…" P17: "…It is a software development everybody should learn…"
A practical and functional program intended for application development	4	P13: "The LightSwitch application is very functional. I should have definitely been taught earlier" P1: "…I believe that is quite a practical program"
I no more have programming fear thanks to this application.	2	P16: "…I wish this training was given earlier, in the freshman year or the second year. Then I could have defeated my fears about my department. At least not only coding would come into my mind when I heard about programming" P10: "…they should realize that via these kinds of programs students gain self-confidence as they themselves can produce something"
Future projects should be emphasized	2	
I do not think that this program will attract much interest in the future	1	
It made me upset to take this course in senior grade	2	
While I had no interest in programming this program increased my desire to develop applications	2	P18: "…Going on this topic, more time should be allocated to the students and projects on needs should be made"

According to Table 17, pre-service IT teachers see RAD tool, as an opportunity and expects it to become more popular in the coming years. In addition to these, with LightSwitch application development, they were not stressed, they were increasing their programming skills and they were enjoying the pleasure of developing products.

To sum up the findings; in the first stage of the research, the current situation of pre-service teachers were determined via *Self Efficacy Scale for Educational Software Development*. Programming was lower than other dimensions and no meaningful difference was found between the grade levels

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

except for "Project Management and Instructional Design" dimension. The reason is thought to be the structure of the CEIT curricula in which project management courses are given in senior grade.

In the second stage of the research, *Computer Programming Self-Efficacy Scale* was applied and it was seen that computer programming self-efficacy of pre-service teachers in performing complex programming tasks were low.

Finally, after the scale implementations, views of pre-service teachers were examined. The satisfaction levels of pre-service IT teachers about the training program were at a high level; moreover, they believed that with the help of this training they attended, they could now develop faster and more effective training software. Majority of pre-service teachers stated that they did not have any difficulty in the training program they attended. The views of pre-service IT teachers about the contribution of this program to themselves were grouped into cognitive and affective levels. Nearly all pre-service IT teachers believe that all CEIT graduates should have a knowledge of rapid application development as well as take course or courses on these applications. Some of the pre-service teachers believe that the 4-year training program in their departments made them gain the skill of developing software that could bring solutions to real life problems whereas some stay neutral and a few do not believe in this situation. Some of the pre-service teachers felt themselves inadequate, some neutral and a very few competent before attending the training program. Whereas after attending the course all pre-service teachers felt themselves confident in programming. Pre-service IT teachers also emphasized that the program was a practical one that should be made a compulsory course in the CEIT curricula as well as being a program which erased their fears about both the department and the field. They also added that this course should have been given in the earlier grades to help them gain the self-confidence and make better studies.

5. Discussion, conclusion and recommendation

The number of undergraduate programs in higher education in the field of ICT has been increasing. However, it can be said that the need for qualified staff is not met in our country and also in the world. It is seen that both public institutions and private sector organizations have serious job advertisements but it is seen and known that educated people who meet this demand are not qualified to meet these needs even though they have graduated from related programs. On the other hand, many countries are in trainings and engagements such as robotics, coding and algorithms for children. The need for teachers with pedagogical knowledge to provide these trainings is also increasing day by day. In this research, it is seen that programming self-efficacy levels of pre-service IT teachers from Computer Education and Instructional Technologies Department, are too low when compared with other domain specific competencies. Findings of Stage I showed that Factor 4 (Programming) average score was the lowest one. In other words pre-service IT teachers indicated that they had difficulties in programming. Furthermore, findings of Stage II were compatible with the previous one. Also knowledge level of the pre-service IT teachers were found to be unsatisfactory in this stage of the study. Problems and difficulties experienced in programming education are given in literature (Robins, Rountree & Rountree, 2003; Tan, Ting & Ling, 2009). This study investigated if RAD training can be used as an icebreaker in order to change Turkish pre-service IT teachers' perceptions towards programming positively. According to analysis results of the data derived from the pre-service IT teachers, the hypothesis is actually confirmed on this study group. When it comes to features of RAD tools, it is obvious that we got maximum efficiency with minimum syntax. Thus, the software developer (pre-service teacher) using the RAD tool feels more comfortable and secure because s/he does not face syntax, remembering and miswriting, etc. very often. A sample expression of one of the participants is *"I wish this training was given earlier, in the freshman year or the second year. Then I could have defeated my fears about my department. At least not only coding would come into my mind when I heard about programming"*. Considering the reflections of the pre-service teachers completing the RAD training, they were satisfied and happy for developing a complete product without difficulty and stress. In this study, Microsoft LightSwitch was used as a RAD tool. Microsoft has included PowerApps this year as a second product to its

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

product family. Microsoft is rallying these tools with the awareness that the business world has to move very fast and that IT solutions are now a reality for businesses that maintain continuity, uninterruptedness and enterprise. It says no institution should allocate long-term coding time for the software; it must take time to analyze the work and reach the conclusion. But of course, developing software to solve problems, creating algorithms and workflows are not expected to be completely solved with RAD tools. People should be aware of these tools, know where to use them, and most importantly, other software technologies should never be overlooked. One of the features of the LightSwitch tool is that it contains the relational database itself. Thus, pre-service IT teachers had the opportunity to combine and use previous knowledge they gained in previous lessons. They were provided with experience in such issues as the need for relationships, the importance of data types, and the need for constraints. Besides, it was quite motivating for the learners to allow the developed software to work as a desktop application and even on the web by changing a value in the settings.

The results showed that RAD tools are icebreakers in the context of the study. In this respect, it is recommended that these tools be included in the higher education programs providing informatics education such as CEIT. The growing need for mobile software has become inevitable in today's world, where people now want to manage everything from their pockets. For this reason, it seems to be beneficial if we are aware of the power of these tools and integrate them by considering how we can benefit both our work and education. We should not give up on the question of what other icebreakers might be.

References

Anastasiadou, S. D., & Karakos, A. S. (2011). The beliefs of electrical and computer engineering students' regarding computer programming. *International Journal of Technology, Knowledge & Society*, *7*(1), 37-51.

Altun, A. & Mazman, S. G. (2012). Developing computer programming self-efficacy scale. *Journal of Measurement and Evaluation in Education and Psychology*, *3*(2), 297–308.

Aşkar, P., & Davenport, D. (2009). An investigation of factors related to self-efficacy for Java Programming among engineering students. *Turkish Online Journal of Educational Technology*, *8*(1), 26-32.

Baser, M. (2013). Attitude, gender and achievement in computer programming. *Middle-East Journal of Scientific Research, 14* (2), 248-255.

Cetin, I., & Ozden, M. Y. (2015). Development of computer programming attitude scale for university students. *Computer Applications in Engineering Education*, *23*(5), 667-672.

Creswell, J. W. (2008). *Educational research. Planning, conducting, and evaluating quantitative and qualitative research*. London: Sage Thousand Oaks.

Daud, NMN, Bakar, AAA, Rusli, HM. (2010). *Implementing Rapid Application Development (RAD) Methodology in Developing Practical Training Application System*, International Symposium on Information Technology, Kuala Lumpur, Malaysia, 15-17 June 2010.

Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, *2*(1), 57-73.

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.

Gençtürk, A. T., & Korucu, A. T. (2017). The effects of Web 2.0 Technologies usage in programming languages lesson on the academic success, interrogative learning skills and attitudes of students towards programming languages. *Higher Education Studies*, *7*(1), 114.

Goadrich, M. (2014). Incorporating tangible computing devices into CS1. *Journal of Computing Sciences in Colleges, 29*(5), 23-31.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Gouws, L. A., Bradshaw, K., & Wentworth, P. (2013). Computational thinking in educational activities: An evaluation of the educational game light-bot. *In Proceedings of the 18th ACM conference on innovation and technology in computer science education* (pp. 10-15). New York, NY, USA.

Hashim, N. M. Z., & Mohamed, S. N. K. S. (2013). Development of student information system. *International Journal of Science and Research (IJSR) 2*, 256-260.

Huaqing, M., & Li, Z. (2011, March). Template-Based Framework for Rapid Application Development Platform. In *Power and Energy Engineering Conference (APPEEC), 2011 Asia-Pacific* (pp. 1-4). IEEE.

Jegede, P. O. (2009). Predictors of java programming self–efficacy among engineering students in a Nigerian University. *International Journal of Computer Science and Information Security (IJCSIS), 4*(2).

Kafai, Y., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan, 95* (1), 61–65.

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior*, *52*, 200-210.

Kalelioğlu, F., & Gülbahar, Y. (2014). The effect of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33–50.

Lau, W. W. F., & Yuen, A. H. K. (2011). Modeling programming performance: Beyond the influence of learner characteristics. *Computers and Education, 57*(1), 1202-1213.

Lee, Y. J. (2011). Empowering teachers to create educational software: A constructivist approach utilizing Etoys, pair programming and cognitive apprenticeship. *Computers & Education*, *56*(2), 527-538.

Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K–8 curriculum. *ACM Inroad, 5*(4), 64–71.

Leung T. (2015) Introducing LightSwitch. In: *Visual Studio LightSwitch 2015*. Apress, Berkeley, CA.

Lohr, L., Javeri, M., Mahoney, C., Gall, J., Li, K., & Strongin, D. (2003). Using rapid application development to improve the usability of a preservice teacher technology course. *Educational Technology Research and Development*, *51*(2), 41-55.

Maltby, John R., & Jan Whittle. (2000) "Learning programming online: Student perceptions and performance." In *Proceedings of the ASCILITE 2000 Conference*. 2000.

Martin, J. (1991). *Rapid application development*. Indianapolis: Macmillan Publishing Company.

Mazman, S. G., & Altun, A. (2013). Programlama–I dersinin BÖTE bölümü öğrencilerinin programlamaya ilişkin öz yeterlilik algıları üzerine etkisi. *Journal of Instructional Technologies & Teacher Education*, *2*(3), 24-29.

Miles, M. B. & Huberman, A. M. (1994*). Qualitative data analysis: An expanded sourcebook*. Thousand Oaks, CA: Sage.

Milne, I., & Rowe, G. (2002). Difficulties in learning and teaching programming-views of students and tutors. *Education and Information Technologies*, *7*(1), 55-66.

Morrison, M., & Newman, T. S. (2001). A study of the impact of student background and preparedness on outcomes in CS I. In *ACM SIGCSE Bulletin* (Vol. 33, No. 1, pp. 179-183). ACM.

Mow, I. C. (2008). Issues and difficulties in teaching novice computer programming. *Innovative techniques in instruction technology, e-learning, e-assessment, and education*, 199-204.

Munassar, N. M. A., & Govardhan, A. (2010). A comparison between five models of software engineering. *International Journal of Computer Science Issues (IJCSI)*, *7*(5), 94-101.

Özoran, D., Cağıltay, N. E. & Topallı, D. (2012*).* Using Scratch in introduction to programming course for engineering students. *In Proceedings of 2nd International Engineering Education Conference (IEEC2012)* (pp. 125–132). Antalya, Turkey.

Üstündağ, M. T. (2017). Could rapid application development tools be used as icebreakers in programming education? *Journal of Human Sciences*, *14*(4), 4586-4605. doi:10.14687/jhs.v14i4.5155

Ramalingam, V., & Wiedenbeck, S. (1998). Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. *Journal of Educational Computing Research*, *19*(4), 367-381.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education*, *97*, 129-141.

Shaw, R. S. (2013). The relationships among group size, participation, and performance of programming language learning supported with online forums. *Computers & Education*, *62*, 196-207.

Tan, P. H., Ting, C. Y., & Ling, S. W. (2009). Learning difficulties in programming courses: undergraduates' perspective and perception. *ICCTD'09. International Conference on Computer Technology and Development* (Vol. 1, pp. 42-46). IEEE.

Yıldırım, A. & Şimşek, H. (2006). *Sosyal bilimlerde nitel arastırma yöntemleri*. Seçkin Yayıncılık.

Yurdugül, H., & Aşkar, P. (2013). Learning programming, problem solving and gender: A longitudinal study. *Procedia - Social and Behavioral Sciences, 83*, 605-610.

Yükseltürk, E., & Altıok, S. (2017). An investigation of the effects of programming with Scratch on the preservice IT teachers' self- efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology*, *48*(3), 789-801.

Zainal, N. F. A., Shahrani, S., Yatim, N. F. M., Rahman, R. A., Rahmat, M., & Latih, R. (2012). Students' perception and motivation towards programming. *Procedia-Social and Behavioral Sciences*, *59*, 277-286.