I.J. Modern Education and Computer Science, **2016, 8, 1-10**
Published Online August 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2016.08.01

Comparing Elementary Students' Programming Success based on Programming Environment

Monika Mladenović
Elementary school Blatine-Skrape, Split, 21000, Croatia
Email: monika.mladenovic@skole.hr

Marko Rosić and Saša Mladenović
Faculty of Science, University of Split, Split, 21000, Croatia
Email: {marko.rosic, smladen}@pmfst.hr

***Abstract*—**In the Republic of Croatia Informatics is an elective course in elementary school for students from 5th to 8th grade. When it comes to programming language teachers can choose between BASIC and Logo. There are a lot of new programming environments for learning and teaching programming like Scratch and also new ways of teaching programming like game based learning. This study compares attitudinal and learning outcomes of 7th-grade students programming in Logo and Scratch. The classes were normal classes, non-extracurricular activities. The questionnaire is used to measure the attitude towards programming and programming languages. The test is constructed to measure learned programming concepts in both compared programming languages Logo and Scratch. Results showed that learning Scratch first can provide a better understanding of basic programming concept for novices in elementary school than Logo.

***Index Terms*—**Programming, Scratch, Logo, elementary school, game based learning.

I. Introduction

Programming is challenging both for educators and students [1] and is considered hard for learning and understanding, especially for novices [2] [3] [4] regardless of age [5]. Most of the elementary students have a negative attitude towards programming [6]. Some research showed that factors such as attitude and motivation towards programming affect the success in learning to program [7]. Although programming is a young discipline because of significant technology progress in last few decades, the way of learning has not changed. Fundamental problems used in teaching programming are still math problems while computers and programming are not perceived anymore just as great calculators used for solving complex math equations. Programming is a way of thinking, not a conditioning skillfulness. Learning about language syntax and basic constructs is comparable to learning about pencils and paper when learning how to draw.

Some studies have indicated that factors as attitude and motivation have a significant influence on success factor when learning to program [7]. "Old school" way of learning to program, without relating to students' interests is not encouraging students for learning programming and computer science. [8] [9].

One of the first programming languages designed for children, used as a tool to learn geometry by mastering problem-solving skills is Papert's Logo [10]. Results of programs execution written in Logo are visually attractive. The problem for novices is typed syntax, especially when it comes to non-English speakers. Scratch is a visual programming language based on Logo philosophy, but the use of drag and drop blocks [11] presented in the form of puzzles eliminates the typed syntax problems, allowing to focus on semantics instead on syntax.

This study compares students' test results in Logo and Scratch and also students' attitude towards programming and programming languages after treatment in school settings. We hypothesized that in the comparison results in Logo and Scratch:

- Students will achieve better results in Scratch than Logo. Therefore Scratch environment provides a better programming environment for mastering basic programming concepts for this age.
- Students attitude towards programming after using Scratch is better.
- Students more like Scratch than Logo environment.

In the next section, the general background and related studies are described, followed by the description of the research methodology used in the third section of the paper. Finally, in the fourth section results and observations based on the data collected are presented and commented, providing answers to the hypotheses presented in the introduction section.

II. Background and Related researches

In the Republic of Croatia Informatics is an elective course in elementary school for students from 5th to 8th grade. In every grade, programming is one of the learning topics. Based on Croatian national curricula [10] teachers can choose between BASIC and Logo for a programming language.

As programming is not popular among children, selection of appropriate programming language can be crucial in motivation and attitude towards programming. At the same time, children should master basic programming concepts that shouldn't be neglected just for motivational factors.

Researchers conducted studies in clubs or summer schools and school settings. Summer schools and clubhouses are learning environments with the informal setting. One of the benefits of informal learning spaces is that youth can explore their interests in a lengthy, uninterrupted span of time [11]. Researchers conducted one of such studies at Harvard's summer school where the goal of the research was to approve students' attitude toward programming. They were using Java after using Scratch programming language. Results showed that children consider Scratch fun, and they had a positive attitude towards programming after using Scratch, a small number of students that didn't like Scratch were mostly students with prior programming experience in procedural programming languages [12]. Members of a Clubhouse near Los Angeles in another research used Scratch for creating games, animations, music videos, etc. Results showed that children were able to master basic programming concepts in Scratch and that Scratch was fun for them. [13] The other study compared the attitudinal and learning outcomes of sixth-grade students programming in either Logo or Scratch in summer school. Researchers found that Logo environment appeared capable of supporting student development of confidence, interest in computer programming, and understanding the loop construct, the Scratch environment provided a relative improvement in learning outcomes for students learning the construct of conditionals [14].

It is questionable if research studies in informal settings are realistic since children are participating voluntarily in clubhouses or schools proving that the motivation for programming of participants is already very high.

Some research studies are realized in more realistic, school setting. One of those showed that Logo improves students' creativity and problem-solving skill amongst grade 5 students in Catholic elementary school near Jakarta [15]. Another research in a classroom setting conducted in Scotland found that by programming games, while programming in Scratch, children can construct games using programming concepts [16].

Scientists conducted a couple of studies in high school in Israel where they used Combined SOLO, and Bloom's taxonomy as part of the research for designing the tools and the later analysis among novices from two classes. Results of the first study showed that student can master programming concepts using Scratch supporting the claims that Scratch can be a viable platform for teaching CS. Despite that, some difficulties have been encountered when teaching initialization, variables, and concurrency. In the terms of the SOLO taxonomy, internalization of observed CS concepts demands at least multi-structural abilities, and, in some cases, even requires relational abilities, while internalization of other concepts can be achieved by local, uni-structural abilities [17]. This research also emphasized possible problems related to programming habits like bottom-up programming or extremely fine-grained programming [18], but on the other hand, scenario-based programming is considered "natural" so it should not be concerning [19]. Researchers conducted the second study in high school in Israel among students belonging to five classes. Most of the students in three classes had previous experience in Scratch (they were part of a previous study [17]) belonging to an experimental group and two classes did not have previous Scratch experience belonging to control group. As a methodology, both qualitative and quantitative methods are used. Quantitative analysis showed that significant difference showed only for repeated execution (loops), and the final test showed a significant result on the highest cognitive of Relational Creating in combined SOLO and Bloom taxonomy. As repeated execution is considered difficult concept in programming for a long time than using Scratch is more than justifying its use. As part of the qualitative analysis, teachers reported increased efficiency of students and doubling of enrollment in secondary school CS. The important report is that the students who learned Scratch were drawn from a larger population, and we might have expected their average grades to be lower. Students reported higher levels of motivation and self-efficacy which should improve retention [20].

In Turkey, the researchers explored the effect of Scratch programming on 5th-grade primary school students' problem-solving skills and found no significant differences, but they underlined possibility of achieving different results in a different context and research designs. They also found that there was a slight improvement in the students' self-confidence in their problem-solving ability which sheds light on the possibility that programming may affect students' problem-solving skills. Just providing the learning environment will not be sufficient to fulfill the need of effective teaching. [21]

Dann et al. [22] organized research at an undergraduate level about the transfer from visual programming language Alice to Java. Results showed that student in the experimental group (from Alice to Java) had significantly better results than students belonging to control group (Java only) during two school years which can approve using visual programming languages as a bridge to real programming languages.

In the Republic of Croatia, the survey is conducted among 1462 8th grade students. The results showed that students from Croatian elementary schools fit into the global trends of decreased interest in programming for women despite roughly equal participation in the elective course of informatics. [23]. Besides that, further data analysis showed that programming is one of the least popular topics in Informatics curricula [24]. Dominant motivations to enroll in elective informatics course are an achievement and extrinsically motivation, confirming trends from other countries, demonstrating that students are not enough intrinsically motivated to enroll in computer science courses. [25].

III. Research design and Methodology

In this section, we propose research questions, describe research context and methodology which includes data collection, participants, research tools, collected data and tests used for data analysis.

A. Research Questions

We hypothesized that in comparison results in Logo and Scratch:

- Students will achieve better results for CS concepts in Scratch than Logo. Therefore Scratch environment provides a better programming environment for mastering basic programming concepts for this age (recognition of instructions, prediction of script execution, loop, nested loop, writing a program).

We assumed that students' attitude towards programming would be better after using Scratch and that student like more Scratch than Logo environment.

B. Research Context and process

This research was conducted during 2013/2014 school year and considered programming novices in 7th grade of elementary school; therefore all students were 13-14 years old.

Teacher, the first author, was the same for both groups. The teacher had two years informatics teaching experience in elementary schools including experience in Logo and BASIC teaching and successfully preparing students for Logo and BASIC programming competitions, four years of computer science teaching experience at the undergraduate level, eight years of professional programming experience and Ph.D. candidate for educational research in CS field. Informatics is an elective course in elementary schools in the Republic of Croatia, so the student chose to enroll in Informatics course. Students were in two classes; one consisted of 10 students while the other consisted of 12 students. All students were novices in programming. Therefore, a pretest was not conducted, instead of that students were compared by math accomplishment because of a positive correlation between math and programming [26].

Tables 1. and 2. are representing the research design.

Table 1. – Logo first group treatment

Logo first group				
week		**Topic**	**New instructions**	**Concepts**
1	**Logo**	Drawing basic shapes	fd, rt, lt, repeat	moving a turtle using basic instructions, loops
2		Drawing shapes	Procedures	Procedures
3		Drawing shapes	Procedures with parameters	Procedures with parameters
Logo posttest				
4	**Scratch**	Aquarium program	forward, left, right, repeat	sprites, concurrency, loops
5		Chasing Ghosts game	if	conditionals
6		Simple ricochet game	communication by messaging, conditional loops	loops with conditionals
Scratch posttest				

Table 2. – Scratch first group treatment

Scratch first group				
week		**Topic**	**New instructions**	**Concepts**
1	**Scratch**	Aquarium program	forward, left, right, repeat	sprites, concurrency, loops
2		Chasing Ghosts game	if	conditionals
3		Simple ricochet game	communication by messaging, conditional loops	loops with conditionals
Scratch posttest				
4	**Logo**	Drawing basic shapes	fd, rt, lt, repeat	moving a turtle using basic instructions, loops
5		Drawing shapes	Procedures	Procedures
6		Drawing shapes	Procedures with parameters	Procedures with parameters
Logo posttest				

One class was learning to program in Logo (Logo first group) in first three weeks of the experiment.

Second class was learning to program in Scratch (Scratch first group) by programming games.

After that, both groups were tested for basic programming concepts understanding (basic instructions, loops). Tasks were almost equal, with the difference only in the programming language used for testing. After first three weeks of lecturing, programming languages between groups were interchanged, and tasks were kept the same as for the other group. After six weeks groups were tested again but using different programming language.

As mentioned in the introduction, Logo is considered first programming language that supports concrete to abstract paradigm, but students still have to struggle with the language syntax. Syntax problem is more highlighted when students are not native English speakers. BASIC language is procedural language, and the main problem for beginners is again language syntax, which is also a more serious problem for non-English speakers. Scratch presents commands in the form of visual blocks with localized titles (there is also a Croatian version), therefore eliminating syntax or language problem. For research purpose, Logo is chosen as one of the languages proposed by national curricula and on the other hand Scratch as visual language appropriate for elementary school children. Both languages are offering a concrete representation of program execution, and Scratch is also offering a concrete representation of instructions in the shape of color-coded puzzles.

In both cases, students were guided. For programming in Logo problem solving of tasks was represented by explaining math background of problems (geometry) on the board and then programming in Logo. Mentioned approach is an “old school” problem-solving teaching method, limited to solving math problems as the only connection to “real life” and that one connection is also abstract. Fig 1. presents examples of programs made in Logo.

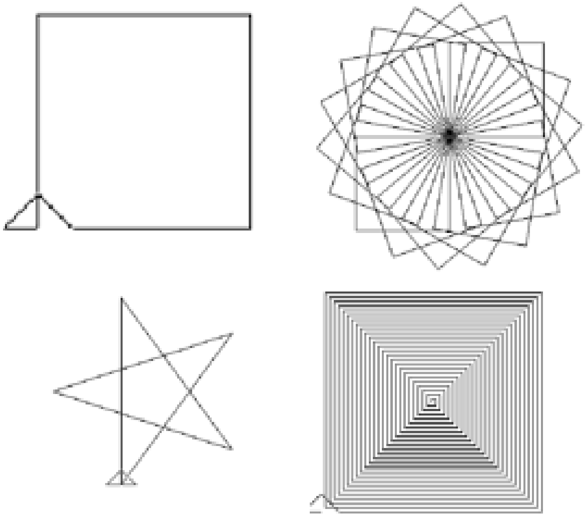


Fig.1. Examples of Logo programs

For problem-solving in Scratch, algorithms are presented as the scenario of games. Students were active in the elaborating scenario and during programming students had to follow the scenario (algorithm). This process is based on connecting problem solving and algorithms with examples from their reality (games) so making programming concepts more tangible to students. Furthermore, concepts of algorithm and programming are distributed, and teamwork is encouraged. Examples of games made in Scratch are depicted in Fig 2.

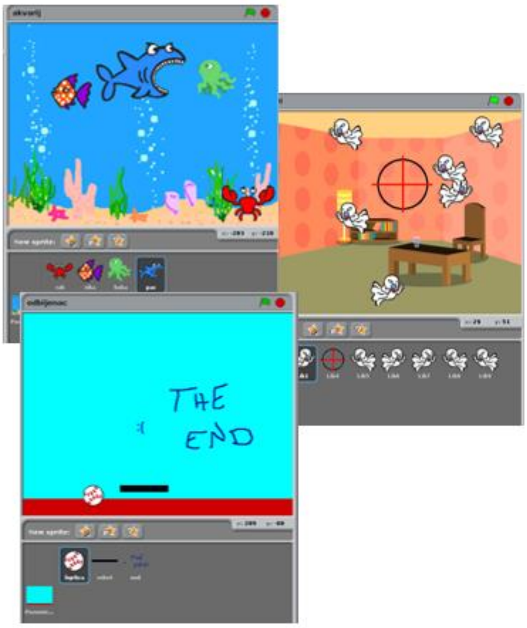


Fig.2. Examples of Scratch programs

Due to the time limit boundaries and official informatics curricula we were in a position to consider basic programming concepts like instructions and loops. Conditionals, communication by using message interchange and concurrency were learned in Scratch but not in Logo programming language. Therefore we could not compare it. On the other hand, procedures and procedures with parameters are learned in Logo but not in Scratch programming language. Therefore those concepts cannot be compared. Results were analyzed by qualitative and quantitative techniques used for triangulation purpose [27] to increase the validity of the findings.

C. Participants

In this research, programming novices in 7th grade of elementary school were studied; therefore, all students were 13-14 years old. The researcher conducted the research during 2013/2014 school years. The teacher was the first author for both groups observed, with two years informatics teaching experience in elementary school including experience in Logo teaching and preparing students for Logo programming competitions, four years of CS teaching experience at the undergraduate level, and also eight years of professional programming experience. The researcher observed students from two classes; one consisted of 10 students while the other consisted of 12 students. All students were novices in programming. Therefore, a pre-test was not conducted, and instead of that students were compared by math accomplishment because of the positive correlation between math and programming [32].

D. Research Tools

The experiment was conducted in 6 weeks of two hours of lectures every week. Two tests have been realized, first after three weeks and the second after six weeks. One class was learning to program in Logo (Logo-first group) in first three weeks of the experiment. The second class was learning to program in Scratch (Scratch-first group) by programming games. After that, both groups were tested for basic programming concepts understanding, tasks were almost equal, with the difference only in the programming language used for testing. After first three weeks of lecturing, programming languages between groups were interchanged, and tasks were retained as for the other group. After six weeks groups were tested again but using different programming language.

For investigation purpose three groups of tools were used:

- Two posttests, one conducted after Logo and one after Scratch
- Attitude survey conducted after six weeks
- Observation by participant, teacher, the first author, as participating observer

Tests are composed of five questions, analogous for Logo and Scratch. The tests were equal for both groups. Besides that, the survey was conducted in both groups. Questions are structured gradually from easier to heavier. Table 3. shows a description of test questions.

Table 3. Description of test questions

Question	Outcome
Q1	Recognition of instructions
Q2	Predicting outcome of execution of script execution
Q3	Predicting outcome of execution of script with loop
Q4	Predicting outcome of execution of script with nested loop
Q5	Writing an algorithm

E. Collecting data

The experiment was conducted in 6 weeks of two hours of lectures every week. Computer science teacher, who is one of the researchers, collected the data in school settings during informatics classes in primary school in Split, Croatia. Observations are also made by the same computer science teacher in students' natural environment without a presence of the third party.

F. Data analysis

The data analysis is using both qualitative and quantitative techniques.

For quantitative analysis, several tests are used. Mann-Whitney test is used to examine if there are differences in math success and also Croatian language success. Shapiro-Wilk test is used to determine normality of posttests results. The chi-square test is used to determine differences between groups for questions. The t-test is used to examine if there are any significant differences between posttest results of the two groups.

Kruskal-Wallis test is used for survey analysis to examine if there are differences in attitude towards programming and programming languages.

Qualitative analysis includes observation during classes and tests, and also analysis of open questions of the survey.

IV. Results and discussion

In this section, we present our results. First, we compared the groups for differences to choose the right tests. Then, we compared results for every question and total results of both groups. As last, we described the findings of the qualitative analysis.

A. Similarity of groups

According to the fact that all students did not have any programming experience, instead of pretest math and Croatian language grades are compared. These courses are basic, besides that there is a correlation between math success and programming.

Mann-Whitney U test indicated that there are no significant differences between groups in Croatian language success (U=64.0, p=.95) and math success (U=41.0, p=.07). Considering the facts mentioned above, we can conclude that students from both groups are equal, so pre-test is not necessary.

B. Distribution of results

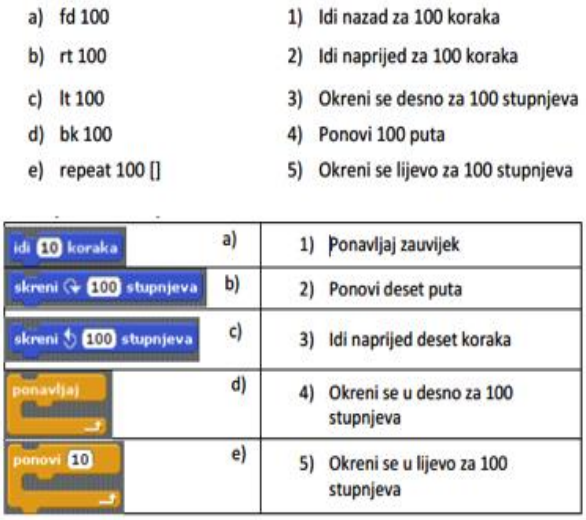
Shapiro-Wilk test is used for testing normality of results. Results showed that all group for both tests had normally distributed data as shown in Table 4.

Table 4. Shapiro-Wilk test results

Group		***df***	***p***
LOGO results	LOGO first	10	,175
	Scratch first	13	,748
Scratch results	LOGO first	10	,056
	Scratch first	13	,842

C. Q1 - Recognition of instructions

After three weeks students from both groups, Logo first and Scratch first, where solving the posttest, each group in the programming language they are learning in previous three weeks. Tasks in the tests were equal with the difference of programming language; therefore, each question is comparable. The test was composed of 5 tasks. The first question was related to recognition of basic instructions as showed in Fig 3.



a) fd 100
b) rt 100
c) lt 100
d) bk 100
e) repeat 100 []

1) Idi nazad za 100 koraka
2) Idi naprijed za 100 koraka
3) Okreni se desno za 100 stupnjeva
4) Ponovi 100 puta
5) Okreni se lijevo za 100 stupnjeva

idi 10 koraka	a)	1) Ponavljaj zauvijek
skreni 100 stupnjeva	b)	2) Ponovi deset puta
skreni 100 stupnjeva	c)	3) Idi naprijed deset koraka
ponavljaj	d)	4) Okreni se u desno za 100 stupnjeva
ponovi 10	e)	5) Okreni se u lijevo za 100 stupnjeva

Fig.3. Q1 (LOGO upper half, Scratch lower half)

Table 5. Frequencies of Q1 answers

		Logo		Scratch		Total
	points	0	1	0	1	
group	LOGO first	1	9	0	10	10
	Scratch first	2	11	0	13	13
Total		3	20	0	23	23

Results showed that all students presented a correct answer to this question in Scratch while three students did not provide the right answer to this question in Logo: one student from Logo-first group, and two students from Scratch-first group. Q1 is graded with 1 point, 0 for the wrong answer, 1 for the correct answer. Table 5. contains frequencies of answers.

Despite that, chi-square test showed that there are no statistically significant differences between groups for Logo Q1 ($\chi(1)=1.14$, $p>.05$) and Scratch Q1 (Q1 is constant for Scratch, so chi-square cannot be computed).

By observing students reaction to both programming languages, it was noticed that instructions in mother tongue are “natural” to students. Instructions in Scratch are natural because they are written as words in their mother tongue, while, for Logo, students had to remember meanings of instruction abbreviations (fd, rt, etc.) which was more difficult to them, especially for weaker students in the English language subject. Logo instructions have two level of difficulties, first are abbreviations of instructions and the second is meaning of those instructions.

D. Q2 - Predicting outcome of execution of script execution

Fig 4. presents Q2 for both test and answer. Beside answer to questions students had to explain their answer. Q3 is graded with 2 points, 0 for the wrong answer, 1 for correct answer with no explanation and 2 points for the correct explanation.

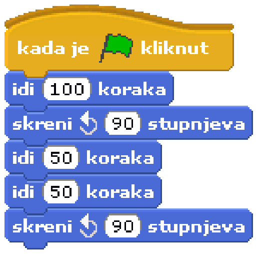
Logo	Scratch	Answers
fd 100 rt 90 fd 50 fd 50 rt 90		a) 0 b) 100 c) 150 d) 200

Fig 4. Q2 (Logo left, Scratch middle, answers right)

Although Scratch-first group had better results for Scratch Q2 than the Logo-first group, chi-square test showed that there are no statistically significant differences between groups for Logo question ($\chi(2)=0.22$, $p>.05$) and Scratch question ($\chi(2)=1.07$, $p>.05$). Table 6. presents frequencies of answers for Q2.

Table 6. Frequencies of Q2 answers

points		*Logo*			*Scratch*		
		0	**1**	**2**	**0**	**1**	**2**
group	*Logo first*	3	3	4	1	4	5
	Scratch first	5	3	5	1	8	4
Total		8	6	9	2	12	9

By analyzing students’ explanations, it was noticed that two students in “Logo-first” group provided the proper explanation but the wrong answer as they were adding additional steps and angles. In Scratch-first group, there were also two students who were adding steps and angles.

E. Q3 - Predicting outcome of execution of script with loop

Fig 5. shows Q3 for both test and answers. Q3 is also graded with 2 points, 0 for the wrong answer, 1 for correct answer with no explanation and 2 points for the correct explanation.

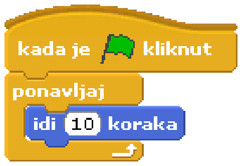
Logo	*Scratch*	*Answers*
repeat 10 [fd 10]		a) 1 b) 10 c) 20 d) 100 *e)* 1000

Fig 5. Q3 (Logo left, Scratch middle, answers right)

Chi-square test showed that there are no statistically significant differences between groups for Logo question ($\chi(2)=3.62$, $p>.05$) and Scratch question ($\chi(2)=0.04$, $p>.05$). Table 7. presents frequencies of answers for Q3.

Table 7. Frequencies of Q3 answers

points		*Logo*			*Scratch*		
		0	***1***	***2***	***0***	***1***	***2***
group	***Logo first***	0	5	5	1	0	9
	Scratch first	1	2	10	1	0	12
Total		1	7	15	2	0	21

Based on the results shown in Table 7. and by analyzing students' explanations, both groups are confident about loop prediction. Usual explanation about the outcome of the script is 10x10 which is correct. However, when it comes to Logo, Logo first group was uncertain about explanation, five students answered correct, but they could not explain why. From the results, it is obvious that "Scratch-first" group adopted well the term loop which they easily transferred to Logo.

F. Q4 - Predicting outcome of execution of script with nested loop

Fig 6. shows Q4 for both test and answers. Q4 is graded with 2 points, 0 for the wrong answer, 1 for correct answer with no explanation and 2 points for the correct explanation.

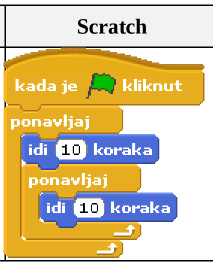
Logo	Scratch	Answers
repeat 10 [fd 1 repeat 10 [fd 1]]	 kada je kliknut ponavljaj idi 10 koraka ponavljaj idi 10 koraka	a) 10 b) 20 c) 100 d) 110 e) 1000

Fig 6. Q4 (Logo left, Scratch middle, answers right)

Chi-square test showed that there are statistically significant differences between groups for Logo question ($\chi(2)=9.44$, $p<.01$) and that there are statistically significant differences for Scratch question ($\chi(2)=1.98$, $p>.05$). Table 8. presents frequencies of answers for Q4.

Table 8. Frequencies of Q4 answers

points		Logo			Scratch		
		0	**1**	**2**	**0**	**1**	**2**
group	**Logo first**	10	0	0	3	1	6
	Scratch first	5	3	5	7	2	4
Total		15	3	5	10	3	10

In the case of nested loop difference is more obvious than in the previous question. As shown in Table 8, no student in the Logo-first group for Logo test did not give the right answer. Although they had five proposed answers, no one answered correctly, not even by luck. The correct answer did not seem plausible to them. The usual answer was 100 steps as they were multiplying values from loops (10x10). Three students from the same group in Scratch test provided the right answer without explanation, and one student presented a correct interpretation. More than half students from Scratch-first group (n=5) for Logo question provided the right answer to this question including sound explanation while three of them answered correctly without providing any explanation. Those are better results than results of the Logo-first group for Scratch question. We can conclude that loops are straightforward to learn in Scratch due to proper visual representation including instructions formulated in the mother tongue.

G. Q5 - Writing an algorithm

Fig 7. depicts Q5 for both test and answers. Q5 is graded with 2 points, 0 for the wrong answer, 1 for correct answer with no explanation and 2 points for the correct explanation.

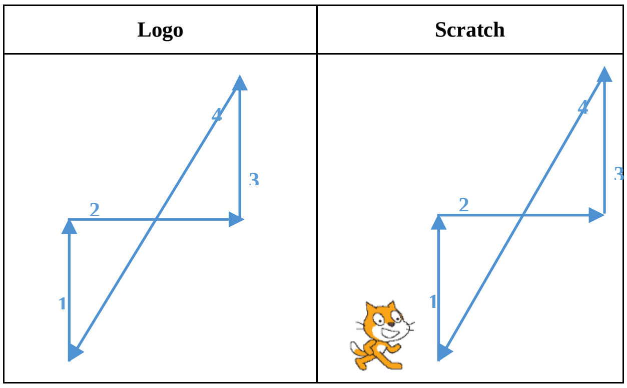


Fig 7. Q5 (Logo left, Scratch right)

Chi-square test showed that there are no statistically significant differences between groups for Logo question ($\chi(2)=2.00$, $p>.05$) and Scratch question ($\chi(2)=2.46$, $p>.05$). Table 9. shows frequencies of answers for Q5.

Table 9. Frequencies of Q5 answers

points		Logo			Scratch		
		0	**1**	**2**	**0**	**1**	**2**
group	**Logo first**	5	1	4	5	1	4
	Scratch first	8	3	2	5	5	3
Total		13	4	6	10	6	7

H. Comparing results between groups

The t-test is used to compare results between groups for both tests. For Logo test, there were no statistically significant differences ($t(21)=-0.67$, $p>.05$). For Scratch test also were no statistically significant differences ($t(14.2)=1.8$, $p>.05$). Table 10. shows the descriptive statistic.

Table 10. Descriptive statistic for tests results

		N	Mean	St. Dev.
Logo	Logo first	10	4,40	2,319
	Scratch first	13	5,08	2,465
Scratch	Logo first	10	5,90	3,178
	Scratch first	13	5,69	1,974

I. Attitude towards programming languages

As shown in frequency in Fig 8., attitude towards scratch is more positive than the attitude towards Logo.

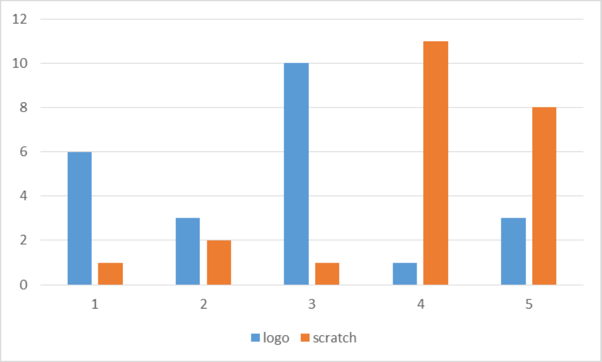


Fig 8. Frequency of answers for attitude towards programming languages

J. Qualitative analysis

The questionnaire contained one open question to collect comments about programming. Students' comments are also showing that Scratch was more fun, interesting, and also, programming games were very motivating for children. They said that Logo was boring, probably because when compared to Scratch, Logo is less visually attractive. Few comments were related to instructions in the English language in Logo which represented difficulties for students. Instructions in their native language are more intuitive thus decreasing frustration with syntax.

Teacher, who was also the observer, noticed non-native language problem during work with students. They had difficulties in remembering instructions in English, and they asked if there are instructions in the Croatian language. Another observed problem was related to syntax. When programming in Scratch students has not experienced the syntax problem mentioned before since instructions are represented as visual blocks in The Croatian language while in Logo they likewise had to struggle with the syntax.

V. Conclusion

Programming is hard, it requires the new way of thinking which is demanding, especially for children and beginners in programming. There are many new visual programming languages designed particularly to introduce programming to children in a more gentle way. One of those languages is Scratch. Benefits of such approach are firstly in emphasizing motivation towards programming. Introducing to programming by using “classic” languages like BASIC reflects in poor motivation. Logo programming language used in this research is somewhere in between classic programming languages and modern visual programming languages like Scratch. In fact, the underlying philosophy of Scratch is Papert's idea of learning to program used for designing Logo programming language.

In Republic of Croatia Informatics (the European term for Computes science in the USA) is an elective course in primary school from 5th to 8th grade. Programming is part of curricula for every grade. When it comes to programming, teachers can choose between BASIC and Logo programming languages. For the purpose of this research, Logo is programming language of choice while Scratch is an alternative. The researcher conducted the experiment in two classes with two approaches, one is “Logo first” and then Scratch introduction through programming simple games and for the second “Scratch first” by programming games and then Logo.

Instead of pre-test we have used a school success of pupils in math, native language (Croatian) and average school success (GPA). After first three weeks first test is conducted in a programming language that pupils were learning, Logo or Scratch and after that classes were learning the second programming language in the same way as the opposite group. Again, after three weeks the second post-test is conducted whit the same tests as in opposite groups. Beside tests after complete treatment survey is conducted in both groups. Test results are compared using Mann-Whitney U and Kruskal-Wallis tests. Results showed that understanding of complex concepts like nested loops is far easier in Scratch than in Logo. There were no statistical differences in other basic programming concepts. When it comes to motivation, the survey showed that Scratch is far more positively accepted than Logo. Students enjoyed working in Scratch.

Besides that research pointed out that only “stronger” students can “easily” understand basic programming concepts in Logo, but most students did not experience those problems in Scratch.

The resulting conclusion based on this research is that Scratch can be used for drawing students to programming and once they are “hooked” the transfer to “real” programming languages is much easier. Also, Scratch is offered in many local languages, like Croatian, which is also an important factor, as children do not have to struggle with instructions formulated in a foreign language so they can be more focused on assignments and programming concepts adoption. Besides mentioned, programming in Scratch is offering many other possibilities for programming assignments instead of just math problems. One of those possibilities is programming games which are very motivational for students since they are digital natives. Learning programming by programming games will be part of our future research.

REFERENCES

[1] Milne and G. Rowe, "Difficulties in Learning and Teaching Programming—Views of Students and Tutors," *Education and Information Technologies,* vol. 7, no. 1, pp. 55 - 66, 2002.

[2] Gomes and A. J. Mendes, "Learning to Program – Difficulties and Solutions," in *International conference on Engineering Education*, Coimbra, Portugal, 2007.

[3] L. E. Winslow, "Programming pedagogy - a psychological overview," *SIGCSE Bulletin,* vol. 28, no. 3, pp. 17-22, 1996.

[4] Robins, J. Rountree and N. Rountree, "Learning and Teaching Programming: A Review and Discussion," *Computer Science Education,* vol. 13, no. 2, pp. 137 - 172, 2003.

[5] M. Guzdial, "Programming environments for novices," in *Computer Science Education Research*, 2004, pp. 127-155.

[6] Violino, "Time to Reboot," *Communications of ACM,* vol. 52, no. 4, p. 19, 2009.

[7] N. F. A. Zainal, S. Shahrani, N. F. M. Yatim, R. A. Rahman, M. Rahmat and R. Latih, "Students' perception and motivation towards programming," in *UKM Teaching and Learning Congress*, 2011..

[8] S. Uludag, M. Karakus and S. W. Turner, "Implementing IT0/CS0 with Scratch, App Inventor for Android, and Lego Mindstorms," in *SIGITE '11: Proceedings of the 2011 conference on Information technology education* , West Point, NY, USA, 2011..

[9] Forte and M. Guzdial, "Motivation and nonmajors in computer science: identifying discrete audiences for introductory courses," *Education, IEEE Transactions on, 48(2),* pp. 248-253, May 2005..

[10] S. Papert, Mindstorms: children, computers, and powerful ideas, New York: Basic Books, Inc., 1980.

[11] J. Maloney, K. Peppler, Y. B. Kafai, M. Resnick and N. Rusk, "Programming by choice: urban youth learning programming with scratch," Portland, 2008.

[12] Republika Hrvatska Ministarstvo znanosti, obrazovanja i športa, "Nastavni plan i program za osnovnu školu," Ministarstvo znanosti, obrazovanja i športa, Zagreb, 2006.

[13] K. A. Peppler and Y. B. Kafai, "From SuperGoo to Scratch: exploring creative digital media production in informal learning," *Learning, Media & Technology,* vol. 32, no. 2, pp. 149-166, 2007.

[14] J. Malan and H. H. Leitner, "Scratch for budding computer scientists," New York, 2007.

[15] C. M. Lewis, "How programming environment shapes perception, learning and goals: logo vs. scratch," New York, 2010.

[16] B. Pardamean, Evelin and Honni, "The effect of logo programming language for creativity and problem solving," in *Proceedings of the 10th WSEAS international conference on E-Activities*, Nanjing, China, 2001.

[17] Wilson, T. Hainey and T. Connolly, "Evaluation of Computer Games Developed by Primary School Children to Gauge Understanding of Programming Concepts," *Interational Journal of Games-based Learning,* vol. 3, no. 1, pp. 93-109, 2013.

[18] Meerbaum-Salant, Armoni and Ben-Ari, "Learning computer science concepts with scratch," in *Proceedings of the Sixth international workshop on Computing education research*, Aarhus, Denmark, 2010.

[19] Meerbaum-Salan, Armon and Ben-Ari, "Habits of Programming in Scratch," Darmstadt, Germany, 2011.

[20] M. Gordo, A. Marron and O. Meerbaum-Salant, "Spaghetti for the main course?: observations on the naturalness of scenario-based programming," in *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education,* New York, 2012.

[21] M. Armoni, O. Meerbaum-Salant and M. Ben-Ari, "From Scratch to "Real" Programming," *ACM Transactions on Computing Education (TOCE),* vol. 14, no. 4, 2015.

[22] Kalelioğlu and Y. Gülbahar, "The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective," *INFORMATICS IN EDUCATION,* vol. 13, no. 1, pp. 33-50, 2014.

[23] W. Dann, D. Cosgrov, D. Slater, D. Culyba and S. Cooper, "Mediated transfer: Alice 3 to Java," *Proceedings of the 43rd ACM technical symposium on Computer Science Education,* pp. 141-146, 2012.

[24] M. Mladenović, Ž. Žanko and S. Mladenović, "Elementary students' attitude towards programming in the Republic of Croatia," in *CIET*, Split, 2014.

[25] Ž. Žanko, M. Mladenović and S. Mladenović, "Students attitude towards informatics curricula," in *ICERI*, Seville, 2014.

[26] S. Mladenovića, Ž. Žanko and M. Mladenović, "Elementary Students' Motivation Towards Informatics Course," *Procedia - Social and Behavioral Sciences,* vol. 174, no. 12, p. 3780–3787, 2015.

[27] White and M. Sivitanides, "An empirical investigation of the relationship between success in mathematics and visual programming courses," *Journal of Information Systems Education,* vol. 14, no. 4, pp. 409-416, 2003.

[28] L. Cohen, L. Manion and K. Morrison, Research Methods in Education, London: Routledge, 2011.

Authors' Profiles

Monika Mladenović is Computer science teacher working in elementary school Blatine-Skrape in Split, Croatia since 2012. She is also an external associate at the Faculty of Science, University of Split (Croatia) at Computer science department on undergraduate and graduate level since 2011. From 2003 to 2012 she was working as a programmer and is still periodically working as a programmer as an external associate. She is a Ph.D. student at the Faculty of Science, University of Split (Croatia). Her research interest includes computer programming teaching and learning.

Marko Rosić is a full professor of computer science at the Faculty of Science and Provost of University of Split, Croatia. His scientific work is related to the information and communication technology applications. Researching possibilities of using agents and the Semantic Web technologies for developing the e-learning systems based on the intelligent tutoring systems is in the center of his interest. He participates in many scientific and technological projects related to the field of his scientific interest. He has authored or co-authored more than forty scientific papers.

Saša Mladenović is vice dean for education at the Faculty of Science, University of Split (Croatia). He teaches some doctoral, graduate and undergraduate courses at the Department of Informatics. He received his Ph.D. in computer science at the Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture in Split (FESB). From 1999 to 2006 he was acting as Technical manager of the Toll collection system department of Ecsat, Croatia – the company responsible for software development of CS group Designer, Integrator, and operator of mission-critical systems, France. His research interests include problems in teaching programming, interoperability, intelligent technologies like ontology and multi-agent systems, especially engineering applications of intelligent technologies. He has authored or co-authored more than thirty scientific papers. He is IEEE member since 1996.