

Using Code Review at School and at the Programming Club

Zuzana Kubincová¹ 

Comenius University in Bratislava, Slovakia
<http://sluzby.fmph.uniba.sk/ludia/en/kubincova1>
kubincova@fmph.uniba.sk

Iveta Demková

Comenius University in Bratislava, Slovakia
Leisure-time Center, Šala, Slovakia
ivtkacs@gmail.com

Abstract

Educational code review is an activity that not only helps prepare future programmers into practice, but also teaches students to work with code in a different way. In the educational settings, activities focused on code review are mainly encountered at universities. In our research, we focused on lower levels of education and in our previous publications we presented the results of using code review at secondary school. Experimentally, we also tried to use the code review activities on the sessions at the leisure-time activities club. This paper provides a description of the research carried out. We compare the outcomes from the club with the results from the secondary school. We also give an overview of the benefits, as well as the problems such activities can bring to the classroom.

2012 ACM Subject Classification Human-centered computing → Empirical studies in collaborative and social computing; Applied computing → Collaborative learning; Social and professional topics → Computing education; Social and professional topics → Student assessment; Social and professional topics → Computing education programs

Keywords and phrases Code Review, Programming, Leisure-Time Activity Club, High School

Digital Object Identifier 10.4230/OASICS.ICPEC.2020.14

Funding This work was supported from Slovak national project VEGA 1/0797/18.

1 Introduction

In school practice, many methodologies are used to teach programming. In our research we focused on code review and the possibilities of its use as an educational activity.

Code review is a technique commonly used in programming practice in software development. It can be characterized as examining the program code [1] and commenting on it by other programmers, usually by colleagues of the program author. The aim of this activity is to find and correct errors in the program or to optimize the program code and thus improve its quality [5]. As several studies have shown, code review can help detect up to 70% of errors in software projects [9, 10]. Therefore, this technique is considered one of the best practices used by professional programmers in software companies. E.g. Google can serve as a good example of a software giant where code review has become an essential part of software development already since the beginning of the company's history [6].

In order to use code review in teaching, this technique needed to be modified. Thus, the so-called educational code review emerged, which many educational professionals have been dealing with recently in their research. When analyzing publications in this area, we found

¹ Corresponding author



several papers describing the application of code review into programming teaching, however, only at the university level [8, 7, 5]. As our target group is pupils at lower levels of education, we focused on them in our research.

2 Code Review in Our Programming Lessons

Since the beginning of the school year 2017/2018 we have been using the code review technique adapted to the school environment at high school, as well as at the leisure club, attended by pupils of different grades of elementary school, students of high school, and several adults as well. Code review conducted in the programming practice is a fairly complex process having its rules and standards. In introducing this activity into teaching, our goal was not to carry it out in the same way as it is common in development of software projects in programmer teams. We try to apply only some of its elements in teaching, to help learners develop their programming skills.

In some of our previous publications [3, 4], we already analyzed the results of the research, which was carried out at a bilingual five-year high school in 2017/2018. The first phase of the research was conducted in the third grade. Later, the next phase of the research was carried out in 2018/2019. The sample comprised some students who participated in the first phase, however, they were now in their fourth year of study.

In the first half of the year, 52 third-grade students participated in the research, whereas the second half had 55 participants of the third-grade. Teaching was carried out following the national curriculum. During the programming classes in the previous school years, the students used to program in Pascal programming language first and afterward they started to learn Python. At the end of the third grade, they were able to work with the timer and create their own mini-games. In the fourth grade, 10 students, participants of the informatics seminar, took part in the research. They mastered functions, lists, strings, text files, and various algorithmic calculations.

We introduced code review to the classroom teaching in two ways: using small re-views and project reviews. The small reviews were to review short programs prepared by the teacher. The students were assigned a short test on paper, consisting of two tasks. To solve the first task they had to find out what the first program would do if it was executed on a computer, to describe and possibly draw its output. In the other task, they had got a short program containing several errors. In the description of the program, there was written what this program would do if there were no bugs. The student's task was to find the errors, mark them and correct them.

In the other assignment type – the project review – the students reviewed longer programs created by their classmates and mutually commented on them.

At the same time, we conducted an experiment at a programming club, in the leisure-time center. Here we employed the first type of code review activity – small reviews. Five people of different ages and with different previous programming experiences participated in the programming club: a pupil of the fourth grade and a pupil of the eighth grade at elementary school, both having previous experience only with programming in Scratch; a freshman at a high school who simultaneously programmed in Python language in informatics classes at school; and two adults who had only a very basic experience in programming in Basic language from their high school years long time ago.

The club sessions were held once a week and lasted for 90 minutes. Our aim was to teach participants the basics of programming in Python. Gradually, the following topics were taught: variables, basic graphical commands (tkinter library), random numbers, for-loop,

functions, mouse click and keyboard events, conditions, text strings, timer, canvas object movement. Thanks to the higher time subsidies at the club it was possible to familiarize the club members also with such programming concepts and topics, which were not taught at the compulsory informatics classes at the high school and only were discussed at an optional informatics seminar.

The code review was involved in the activities of the club after explaining and practicing each topic. Participants reviewed two given programs within 10 minutes (5 minutes each). As with the small reviews at high school, one of the tasks was to find out what the program would do and the other one to look for errors in the given code. Although the regular club lessons proceeded in a looser style and the participants could cooperate and consult each other, this was not the case during the reviews. This activity was perceived as a “test without any grade”, so everyone had to deal with it by themselves.

3 Programming Club Results

Every small review referred to the last topic taught. The first review consisted of tasks dealing with the for-loop, the second review covered the functions, the third one addressed the conditions, the fourth one the timer, and the fifth one moving the canvas objects.

The following figures depict how small reviews looked like – there is an example of one type of task in Fig. 1 and the other type of task in Fig. 2.

Club participants were evaluated by points for solving these tasks. A maximum of 5 points could have been earned for each code review. For the first task, in which they were to write what the program would do, they could get 2 points, in the second task, where they were supposed to find and correct errors, they could get 3 points.

The results of the small reviews at the programming club are shown in Table 1. Most reviews were made by only four out of five participants, as one of them did not attend the club meetings regularly.

Find out what this program would do if it was executed. Describe it verbally and draw a picture as well.

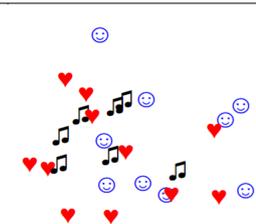
```
import tkinter
from random import *
c=tkinter.Canvas(width=600,height=600,bg="white")
c.pack()

def drawing():
    x = 30
    y = 50
    v = 30
    for i in range(0,6):
        if i%3==0:
            c.create_rectangle(x,y,x+v,y+v,fill="blue")
        elif i%3==1:
            c.create_rectangle(x,y,x+v,y+v,fill="red")
        else:
            c.create_rectangle(x,y,x+v,y+v,fill="yellow")
        x = x + v
    drawing()
```

■ **Figure 1** Example of a code review task (type 1).

14:4 Using Code Review at School and Club

The following program should draw something similar to the picture on the right. However, there are several errors in the program. Find and correct them all.



```

import tkinter
from random import *
c=tkinter.Canvas(width=600,height=600,bg="white")
c.pack()

def tap(event):
    x = range(50,550)
    y = range(50,550)
    if event.keysym = "h"
        c.create_text(x,y,text="♥",font="Arial 30",fill="red")
    elif event.keysym = "s"
        c.create_text(x,y,text="☺",font="Arial 30", fill="blue")
    else
        c.create_text(x,y,text="♪",font="Arial 30",fill="black")

c.bind_all("<Key>", tap)

```

■ **Figure 2** Example of a code review task (type 2).

In the first review, none of the club participants earned any point. No one was able to figure out what would be the result of the first program or find and correct the errors in the second program. The problem was probably in completely new types of tasks they had never encountered before.

The situation changed significantly in the second review when two participants even scored full points and the remaining two earned four and three points. Participants also achieved very similar results in all other rounds of code review. The best results were achieved in the last round – two participants scored full points and the other two lost only one point.

The attitude of the club participants to this activity was explored based on the observation of their work at the club sessions and on the personal interview with them. Their reactions during the interview were positive. While watching their work, we noticed that they often were able to work out both tasks in less time than was available.

■ **Table 1** Code review outcomes at the programming club.

Participant	Gender	Code Review 1	Code Review 2	Code Review 3	Code Review 4	Code Review 5
4th-grader	male	0	5	4	3	4
adult	male	0	5	5	5	5
8th-grader	male	0	4	5	4	5
adult	female	0	3	1	5	4
high school student	male	-	-	4	-	-

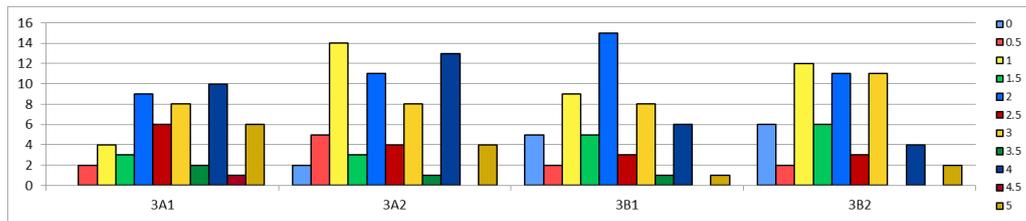


Figure 3 The raw score for all review rounds by groups.

4 Comparison with the Results at High School

Our previous research at high school was conducted in two phases. In the first phase, the research sample consisted of students of two classes in the third grade, divided into four groups (A1, A2, B1, and B2). Although the code review activities involved small reviews as well as project reviews, since we want to compare the results with the results attained at the programming club, we will only present the results of high school students from small reviews. These were carried out at high school and later on in the club in the same way – in five rounds. The programs reviewed by the students and club participants were also the same.

The best results in this activity in high school were achieved by the A1 group, in which there were no reviews with zero rating and three students scored full score in two code review rounds. The A2 group took second place. One student in this group got the full score in two small reviews and two other students in one review. Only one student received a zero rating in two review rounds.

Class B was less successful than class A. In both of its groups, there were 4 students with zero score in one small review and in the B2 group another student has got zero rating from two reviews. Only one (B1) and two (B2) students earned the full score, however, from just one review.

The summary results for all rounds of code review by groups are depicted in Fig. 3. The chart shows the number of students in each group who have earned the appropriate score throughout all review rounds.

After the last review round, students admitted that while these “small tests” were difficult for them, they helped them learn to look for errors in the program and think differently about the program code. So far in the informatics classes, they have only encountered the approach: “I have a problem, my task is to code the program to solve it”. During the code-reviewing, they came to the opposite side: “I have a program code, my task is to find out what the program is doing”.

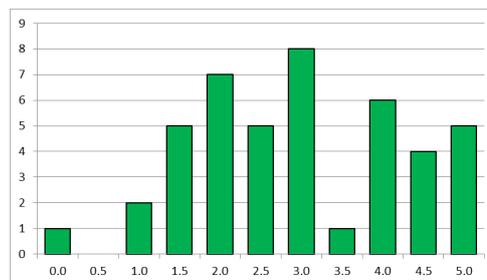
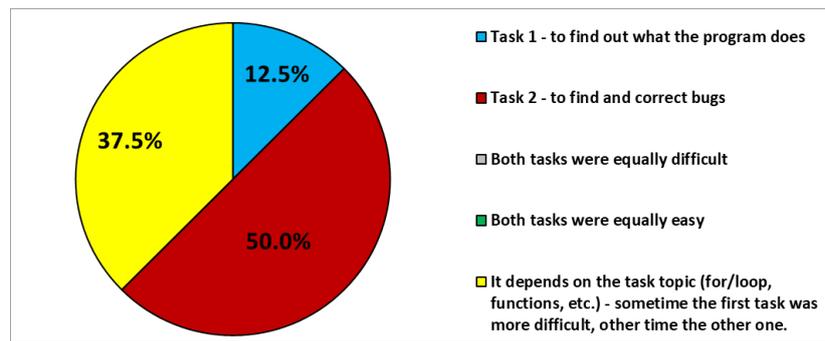


Figure 4 The raw score of fourth-grade students.

14:6 Using Code Review at School and Club



■ **Figure 5** Answers to the question “Which of these two task types did you find more difficult to solve?”.

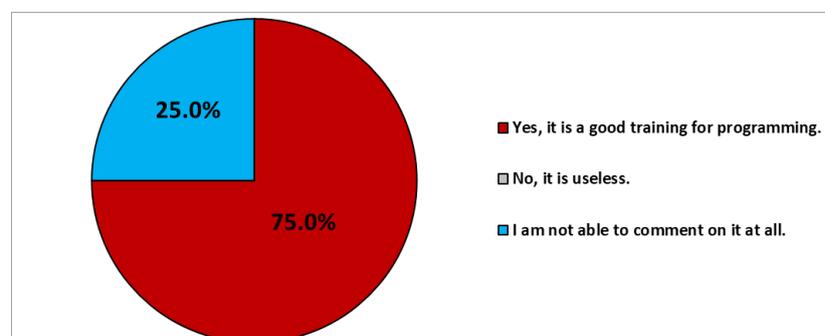
In the following school year (phase two), we tried to verify the previous research results with fourth-grade students who passed small reviews in their third grade. They got programs for small reviews that were identical to those from the previous year. However, the results did not show any significant facts to suggest that students improved in finding errors and finding out what the program would do if it was executed. The raw score of fourth-grade students is shown in Fig. 4.

The detailed results from the third and fourth grades can be found in the thesis of one of the authors [2].

An analysis of the results of the small reviews in both the third and fourth grades shows that the first task type in which students were supposed to find out what the program would do came generally off better than the other one. This finding was confirmed by the fourth-grade students in their responses (Fig. 5) to the questionnaire administered after all the code review activities were completed.

Their opinion on the use of such types of activities in programming teaching was also surveyed by the question, whether they would include such types of tasks in the programming teaching in other grades or schools (Fig. 6). Most students responded positively, saying that this is good training in learning to code. The remaining students did not know how to comment on the question; no one answered negatively.

It is not quite straightforward to compare the results obtained in the school environment with those of the leisure-time club. The programming club is mainly attended by people who are really interested in coding and are therefore showing better results. On the other hand, informatics classes and programming at high school are mandatory subjects, no matter



■ **Figure 6** Answers to the question “Would you incorporate the tasks like this into programming teaching also in other grades/schools?”.

what is the relationship of students to these subjects. In addition, in most schools, the informatics is taught once a week one teaching hour (i.e. 45 minutes). A new topic is often presented in one or two lessons. In the club, each new topic was also explained and trained during one or two club sessions, each session lasting 90 minutes. During this time, the participants have solved considerably more tasks than the students in the high school research did. Another difference is in the size of the group taught by the teacher at school or by the instructor in the club. The leisure-time clubs are usually attended by a much smaller number of participants than the number of students in the classroom, making it easier for the instructor to implement the individual approach method.

The results of experimental research at the programmer's club not only confirmed the anticipated findings that its participants, who were interested in programming and disposed to learn programming, were willing to spend more effort and time to learn to code but also showed that such individuals can quickly accept also less traditional educational activities and show very good results in them. In our case, after their first experience with code review, they understood what was expected of them and improved their reviewing skills both in identifying what the program was doing and in detecting and fixing errors.

5 Educational Code Review Benefits and Problems

Implementation of code review in teaching is not entirely trouble-free and brings with certain complications. In the school environment, the lack of time can be a problem. Implementing code review activities in the form of small reviews requires regular reserving of part of the lesson, which may not be easy as the teacher is supposed to explain and exercise with students quite a lot of topics according to the national curriculum, while the time subsidy for informatics is 45 minutes a week. In addition, since this is a type of activity that students do not usually encounter at school, it is necessary to explain in detail at the beginning what they are expected to do, to show them sample solutions to similar assignments, or to train them for the reviewer role. Again, all of these activities require extra time. However, they are really necessary when reviewing larger programs, such as projects, and certainly also very helpful in the case of small reviews.

Considering the involvement of code review activities at the programmers' club, the time constraints are eliminated as the club has both a larger weekly time subsidy and a looser schedule of activities.

Code reviewing also brings certain benefits to programming teaching. In this way, the teacher gains a broader view of the student, learns how they are doing with programming, but also how they can read somebody else's code, understand it, find out what is the result of the program, find mistakes in it and correct them. The student can get an alternative view of solving problems often the same ones they have already solved themselves, by which they basically learn to program in another way, learn to understand other people's programs, correctly describe mistakes and also develop their communication skills. These and other benefits of using code review in teaching programming, whether at school or at a club, will be enhanced if such type of activity is used to review longer program codes, especially if the students themselves are the authors (see e.g. project reviewing presented in one of our previous publications [4]).

6 Conclusions

Programming practice, research from foreign universities, and also our research show that these activities can gradually improve programming skills. We observed improvement in the quality of the program code even in the case of beginners in programming. Therefore,

in spite of problems with the time subsidy of the subject, we think that teachers should incorporate similar activities into the teaching of programming so that the students gradually get used to them and as soon as possible, can use the benefits these activities offer them.

When employing such activities, it can be useful for the teacher to note during the lesson what mistakes are made by the students in creating programs, what error messages they often encounter, etc. Based on these observations, the teacher can incorporate new elements into programming teaching, for example, also through the code review. This way, it can also be found out whether the student can think about the program, understand it and visualize in abstraction what the program will do and describe it afterward.

It is advisable if activities of this type are carried out with students regularly and especially in the long term. This, too, can help them get deeper into programming and build proper programming habits right from the start.

Our research is currently continuing at the programmer's clubs in the leisure-time center. Several participants from the last year's club continue to attend our club sessions devoted to programming in Python. There is also a new group of beginners learning to program in Python.

With the past and new members of the club, we continue to carry out code review activities. Our goal is to conduct the research on a larger sample so that we can verify previous results and better understand the potential of educational code review in learning to program at leisure-time clubs and its portability to teaching at school.

References

- 1 Barry W. Boehm. Software engineering economics. *IEEE transactions on Software Engineering*, SE-10(1):4–21, 1984. doi:10.1109/TSE.1984.5010193.
- 2 Iveta Csicsolová. Code review v informatike na strednej škole. Master's thesis, Comenius University in Bratislava, Slovakia, 2019. Diploma Thesis, in Slovak.
- 3 Zuzana Kubincová and Iveta Csicsolová. Code review in high school programming. In *2018 17th International Conference on Information Technology Based Higher Education and Training (ITHET)*, pages 1–4. IEEE, 2018. doi:10.1109/ITHET.2018.8424617.
- 4 Zuzana Kubincová and Iveta Csicsolová. Code review at high school? yes! In *International Conference in Methodologies and intelligent Systems for Technology Enhanced Learning*, pages 115–123. Springer, 2019. doi:10.1007/978-3-030-23884-1_15.
- 5 Zuzana Kubincová and Martin Homola. Code review in computer science courses: Take one. In *International Conference on Web-Based Learning*, pages 125–135. Springer, 2017. doi:10.1007/978-3-319-66733-1_14.
- 6 Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko, and Alberto Bacchelli. Modern code review: a case study at google. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, pages 181–190, 2018. doi:10.1145/3183519.3183525.
- 7 Mason Tang. *Caesar: a social code review tool for programming education*. PhD thesis, Massachusetts Institute of Technology, 2011.
- 8 Deborah A. Trytten. A design for team peer code review. *ACM SIGCSE Bulletin*, 37(1):455–459, 2005. doi:10.1145/1047124.1047492.
- 9 Hidetake Uwano, Masahide Nakamura, Akito Monden, and Ken-ichi Matsumoto. Analyzing individual performance of source code review using reviewers' eye movement. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, pages 133–140, 2006. doi:10.1145/1117309.1117357.
- 10 Karl Eugene Wiegars. *Peer reviews in software: A practical guide*. Addison-Wesley Boston, 2002.