Olympiads in Informatics, 2022, Vol. 16, 43–53
© 2022 *IOI, Vilnius University*
DOI: 10.15388/ioi.2022.05

Initial Learning of Textual Programming at School: Evolution of Outreach Activities

Martina LANDMAN, Gerald FUTSCHEK,
Svetlana UNKOVIC, Florentina VOBORIL
TU Wien, Institute of Information Systems Engineering
Favoritenstraße 9-11, 1040 Vienna, Austria
e-mail: martina.landman@tuwien.ac.at, gerald.futschek@tuwien.ac.at,
svetlana.unkovic@tuwien.ac.at, florentina.voboril@tuwien.ac.at

Abstract. Learning programming is at least to some extent part of school curricula in nearly all countries. But in practice there is still a shortage of teachers that got a relevant education in computer science and, in particular, computer programming in almost all countries. Another challenge of teaching programming in schools is the heterogeneity in prior education and programming knowledge of students. This ranges from no prior knowledge to deep experience. We show in this paper how we outreach from our university to teachers and school classes using learning materials and personal support. We also show three stages in the development of our outreach activities and argue why we needed to evolve to the next stages.

Keywords: learning programming, outreach to school, MOOC, processing, workshops.

1. Introduction

Computer programming is not new at school and is part of school curricula in most countries at least to some extent (Bocconi *et al.*, 2022). The use of text-based programming languages is part of the curriculum, especially at the upper secondary level. There exists already a wide variety of online materials and programming courses that are suitable for use at school.

But in practice there are several reasons why in many school classes a satisfiable programming education cannot be given:

1. Engagement of teachers without informatics education due to lack of informatics teachers.
2. Further education of in-service teachers of other subjects is often hard to achieve.
3. The programming environments for textual programming are often overly complex.

4. Teaching programming needs a lot of effort: posing of appropriate tasks, assessing of student programs, reacting to students' problems in writing computer programs.
5. Need of tasks for students of different programming skills levels.

Lack of educated informatics teachers is a major problem in informatics education at school. A major challenge is to educate enough informatics teachers, see (Webb *et al.*, 2017). Most of the countries allow non-informatics teachers to teach programming and other computer science topics. These teachers are usually not experienced in computer programming. They need additional support.

Therefore, our requirements for teaching support in the field of textual computer programming are:

1. Use of a programming language that is easy to learn.
2. Provision of teaching and learning materials: videos, tasks, explanations, tests.
3. Personal support for teachers.

So, our approach is to support teachers in teaching initial programming by personal support and by adaptable teaching material. We describe in this paper three stages of concepts to convey our concept and materials to school classes. We started with use of a MOOC, moved on to a flexible adaptable online programming course and finally provided introductory workshops at school classes.

2. A Programming MOOC for First Year University Students as Basis

When we decided in 2020 to outreach to schools with a textual programming activity, we had already developed in 2018 a MOOC that was based on the programming language Processing, a variant of Java with high potential of learning textual programming. The primary target group of this MOOC was beginners at university level. This MOOC was successfully used in pre-university courses to prepare potential students for MINT studies. A detailed description of concept and structure of this MOOC can be found in Wetzinger *et al.* (2018). Although we implemented some cooperative tasks, the typical use of this MOOC was in self-study during summertime by the already enrolled students.

During construction of this MOOC, we had already a potential use at school level in mind and therefore expected from the users only such pre-knowledge, which is standard pre-knowledge in upper secondary schools. The MOOC consists of two parts and is still available at the public MOOC platform iMooX (iMooX Processing course). All the ten chapters contain several programming tasks, for all of them are also solution programs available.

We advertised this course from the very beginning also for use in school classes, but the use was limited to very experienced teachers that were able to integrate this educational resource into their teaching praxis.

We observed the following barriers for a wider use:

- The teachers cannot observe the learning activities of their students on the iMooX platform, they therefore used an additional platform or asked for copies of the course.

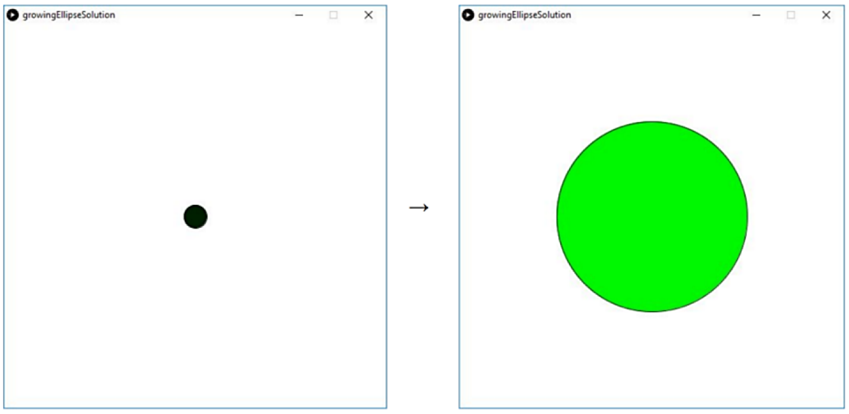
- All provided tasks had already solutions, there was a need for additional tasks for homework and tests.

To demonstrate the transition of our activities we show how a selected task changed from phase to phase.

We give an example of an exercise of the MOOC involving graphics and simple animation, where students must adapt a given code:

The goal of this assignment is to introduce first year university students to the use of control variables and the possibility to bound their value with the modulo operator. This kind of task is intended for interested pre-university students and does not involve hints for a possible solution.

By executing the program (L_Animation_UE2.pde), after your successful adaption, a small black circle should be displayed, which continuously grows to a large green circle. As soon as it has reached its maximum size of 255 it should start again as a small black circle with size 0 and grow again continuously.



Your task now is to add at the marked position (//TODO) an appropriate assignment to the variable `colorIntensity`.

```
int colorIntensity = 0;
void setup(){
  size(500, 500);
}

void draw(){
  background(255);
  fill(0, colorIntensity, 0);
  ellipse(250, 250, colorIntensity, colorIntensity);

  //TODO
}
```

Fig. 1. MOOC-Task including graphics and animations.

3. Online Programming Course for School Classes

We went one step further and wanted to reach a larger and younger target audience with our online materials from the MOOC. As already mentioned, the preparation of the content and supervision in the classroom represents an enormous amount of work for teachers in schools. That is why the idea came up to adapt the existing MOOC contents to students and support teachers with and without programming experience in their computer science lessons. A student-friendly Moodle course for school classes from the 9th grade was created, which is divided into the following fourteen teaching units:

1) Introduction & Basic Figures.
2) Variables.
3) Animations.
4) Characters and strings.
5) Truth values and colors.
6) Logical operators.
7) Branches.
8) While-loops.
9) For-loops.
10) Subprograms.
11) Troubleshooting.
12) One-dimensional arrays.
13) Two-dimensional arrays.
14) Recursion.

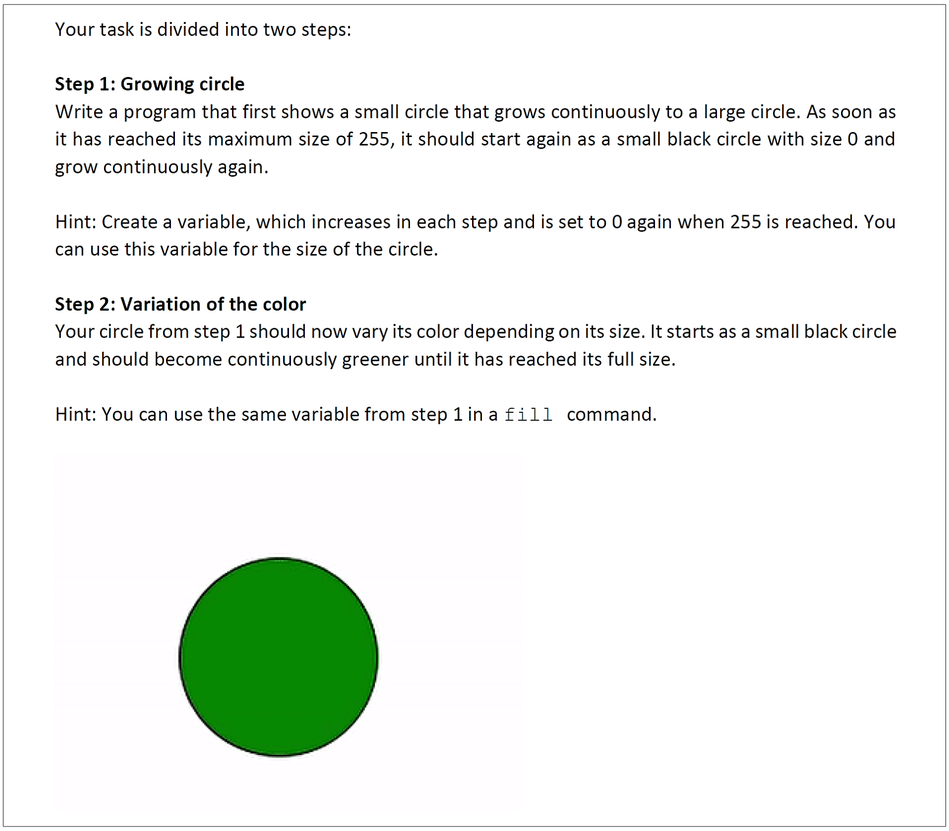
Many in-service teachers are not well trained in teaching programming to their students because that was not part of their education in the past. Due to the massive lack of computer science teachers, schools employ either people who are working in companies outside school and have great programming experience but therefore less pedagogical and didactical knowledge or no-specialist teachers. These non-specialist teachers studied different subjects and additionally took a short course regarding informatics in the past so that they are allowed and able to teach the basics. Unfortunately, many of them show problems in programming because of the missing programming experience. The adaption of this course was intended to help and support those teachers. Another reason why we chose the target group from 9th grade upwards to pre-university school is the mandatory programming part in the school curricula in Austria.

Like the MOOC course, each lesson is dedicated to a programming concept. In contrast to the MOOC for first-year students, the students do not work on a project, but rather work on shorter and simpler exercises with the aim of motivating the learners with small moments of success and developing joy in programming. The content is consolidated with the help of explanatory videos, scripts, cheat sheets, self-examinations and (further) exercises. One of those exercises is stated below.

This exercise is based on the exercise, presented at the previous section, but it was adjusted to the new target group. It uses simpler language and it is divided into two sub steps. In contrast to the version before, the description of the exercise is embedded in Moodle and the resulting program is shown as animation.

Students also have the opportunity to ask questions in the forum if needed. It must be said that the teachers have access to all tasks and their solutions. Depending on the teaching style, the teacher can make the solutions available to the students or discuss specific programs with them.

Since different motives play a role in taking part in the course, the teachers can basically freely design the teaching units and contents. Some classes, for example, have already gained programming experience and only want to learn a new programming language, and others not at all. This then of course rubs off on the course of the lesson and is crucial to how well you can let the students work independently. Nevertheless, we offer a possible process for teachers, as well as a possible assessment scheme. Our idea of the teaching units corresponds to a flipped classroom concept (Sobral 2021). The learners work on the content via videos, scripts and self-examinations from home and can then work on the programming examples in computer science lessons. The computer science class, we could say, acts as a kind of test laboratory. Additionally, teachers have insight into the processing progress of each learner via the Moodle course. In this



Your task is divided into two steps:

Step 1: Growing circle
Write a program that first shows a small circle that grows continuously to a large circle. As soon as it has reached its maximum size of 255, it should start again as a small black circle with size 0 and grow continuously again.

Hint: Create a variable, which increases in each step and is set to 0 again when 255 is reached. You can use this variable for the size of the circle.

Step 2: Variation of the color
Your circle from step 1 should now vary its color depending on its size. It starts as a small black circle and should become continuously greener until it has reached its full size.

Hint: You can use the same variable from step 1 in a `fill` command.

Fig. 2. Online-Course task, which includes animations and hints.

way, the teachers can see the difficulties that the students encounter and this can help to develop a better understanding to their problems. It is also possible to involve employees of the TU Vienna in the school classes. The further the school classes progress with the topics, the more difficult the exercises can become. It may make sense to ask for additional help.

As already mentioned, there are no restrictions on who uses the course. Both teachers with and without programming experience can access the course. From experience, however, most teachers without previous programming experience usually do not trust themselves in teaching the contents of the Moodle course. That is why we offer a slimmed-down version in form of a workshop.

4. From an Online Course to a Workshop in School

After developing and redesigning the MOOC to an Online Course format for School classes, both courses are still available.

- The MOOC, which can be used by private individuals and is accessible via the iMooX platform (registration required). It is split up in two parts.
- The school course with fourteen units, which can be redesigned by the teacher and offers the possibility of student assessment.

The second version, which is more attractive to teachers, is not available through a self-subscription and must be requested through a request form on the same website. Students can also be inserted into this course. However, for this the teacher must send us the students' data so that we can insert them. We chose this format in Moodle to get in touch with every teacher and have the possibility to give them a separate "room" for their teaching with chances for assessment.

In the process, we encountered several difficulties and uncertainties on the part of the teachers. Teachers were unsure about data protection. As the student data is entered into our platform, the parents of the children must agree. There are schools that already cover this case by having the parents sign when the students are admitted to the school. Thus, for some teachers this is not a problem, but for others it is. Therefore, it was decided that the course would be unlocked for teachers to copy. This means that teachers can copy the course and paste it into the school's own Moodle instance.

This leads us to another problem: the transparency of the use of the course. As soon as teachers copy the course into their own platform, we naturally lose the number of students who use the course and who have been reached.

From a survey in autumn 2021, we were able to find out that 93 students were reached. We assume that considerably more students were reached than the survey recorded, as only seven teachers completed the survey, but a total of 17 courses were created. Extrapolating this would give a notional number of 225 students reached. Although 17 new courses were created for teachers in 2021, only one course was created in the first quarter of 2022.

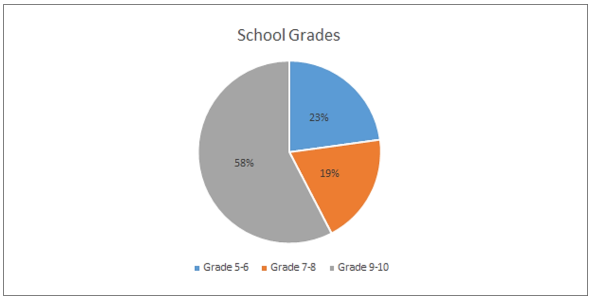


Fig. 3. School Grades in which the online course was used in 2021.

This leads us to the third problem: few interested parties. The decreasing number of interested parties is probably related to the structure of the course, which is difficult to integrate into face-to-face teaching. In the school year 2021/22, no official Distance Learning has been done so far, so it takes a lot of effort for teachers to integrate the content into the classroom. Of course, they are free to use and adapt the content, but many do not have the time.

For these reasons, we have gone one step further and redesigned the course again, consisting of fourteen units, so that you can book a short "Programming with Processing" workshop, taught in 2–4 units, which means 1–2 double lessons in School – 100 Minutes each. For that purpose, we go with our team into schools and do a "Programming with Processing" workshop there in the class. The teacher is assisting our trainers. Afterwards it is up to the teacher if he or she wants to continue with the topic and uses the material from the online course.

From our survey in autumn 2021 we learned, most of the teachers use the course in the 9th to 10th grade. This was the reason, why we designed the workshop for this grade.

5. Programming Workshop in Schools

The following adaptations to the previous version were made:

- From our survey in autumn 2021 we know that the last four units where never done by the teachers in school. Only two teachers did it further than the first two units. So, we decided to make a course with the most essential parts of programming for the first time. The programme was taken (partly) from the first three units of the course. It deals with the most essential contents that are necessary when programming for the first time. The students learn about branching and loops. They learn the structure of a programme.
- The parts that were available as explanatory videos in the original course are now taken over by one of our trainers. This change was made because of the presence workshop format.
- It is taught in a team-teaching format, with students taking turns working together and then doing free exercises.

As before a circle is moved by the mouse pointer on the screen. Now additionally, the circle should be displayed on the left side in green colour and on the right side in red colour. This has the effect that the colour changes whenever the middle line is crossed.

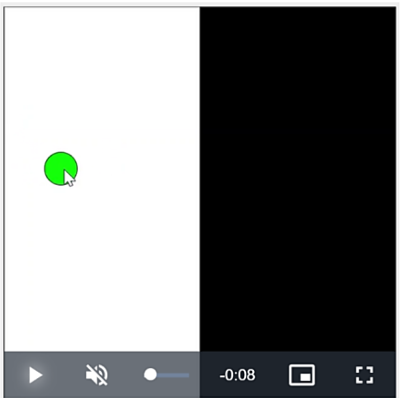


Fig. 4. Task in school workshop, including animations.

- The exercises were designed in such a way that there is always one compulsory task and several additional voluntary exercises. In the pilot test, the weakest students only completed the compulsory task and the most talented completed one or two of the additional tasks.

After a task that allows a circle to follow the mouse pointer on the screen the following task may be given: Fig. 4.

Compared to the exercises before, we included more interactive tasks, which e.g., depend on the position of the mouse pointer. This was especially done to increase the motivation of the students. Also, it fits better to the target group of school students who expect game-like interactions.

The time schedule of the workshop looks as follows:

Part 1
Content
Welcome, introduction and overview
Processing: The programming environment ● Processing surface ● Coordinate system ● Comments ● Difference between setup() and draw()
The first program ● typing of commands together
Explanation of terms: commands, parameters and functions ● Colors (with color selection) ● Importance of the command sequence

Exercises about basic figures
Predefined variables ● height, width ● mouseX, mouseY
Exercises about predefined variables

Block 2
Content
Repetition of contents from block 1
Use of predefined variables in branches new predefined variables: mousePressed, keyPressed, key
Exercises about simple comparisons
Composite comparisons
Exercises about composite comparisons

We piloted the adapted material in two different 9th grade classes with each one week between workshop part 1 and 2. We had one trainer, one teacher assisting and an additional team member for observing and assisting. The conclusion was:

- After the piloted version we had to adapt the schedule to fit better.
- The exercises that were chosen where on a proficient level: The weakest students were able to fulfil all the mandatory tasks in time. This was a good resume for us. The best students did not all of the voluntary tasks, so we have a bit room upwards, if we have an extremely talented group next time.
- Regarding to the teacher, the class gave more attention to the external trainer and where more quiet than usual. This is another opportunity to teach effectively programming in schools when there are external people and not the usual teacher.
- A problem which we faced in the second part of the workshop, one week after the first part, where students that missed the first part. We installed a buddy-system, where one student helped the other to catch up with the whole group.
- One student was extremely interested in programming. Afterwards the class teacher told us, that this student is not interested at all in the “normal informatics lessons stuff,” like using office software.

6. Summary

We showed in this short report how we changed our offer to schoolteachers to support them in teaching initial computer programming. We started with a MOOC for learning Processing consisting of ten chapters, addressing potential novices at University MINT studies. To address students at schools and to support their teachers we adopted this course in a way that we added many tasks and the tasks were made easier understandable. The result was a course with fourteen chapters that fit into usual school lectures.

The last step of adaption was made to support teachers with less experience in teaching programming. We created and offered a workshop of two parts à 100 Minutes that conveys all necessary programming skills to start programming animations. We learned that it is absolutely necessary to understand and fulfill the needs of the target group. Then it is possible to teach programming at various levels of the target group. We are still observing and collecting experiences to improve the course in the future, it is an ongoing process.

References

Bocconi, S., Chioccariello, A., Kampylis, P., Dagienė, V., Wastiau, P., Engelhardt, K., Earp, J., Horvath, M.A., Jasutė, E., Malagoli, C., Masiulionytė-Dagienė, V. and Stupurienė, G. (2022). Reviewing Computational Thinking in Compulsory Education, Inamorato Dos Santos, A., Cachia, R., Giannoutsou, N. and Punie, Y. editor(s), *Publications Office of the European Union*, Luxembourg, ISBN 978-92-76-47208-7 (online), doi:10.2760/126955 (online), JRC128347.

eduLAB online course for school classes: `https://edulab.ifs.tuwien.ac.at/programme/onlinekurs-fuer-schulen/`

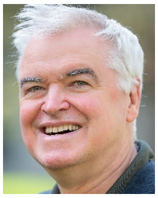
iMooX Processing course: `https://imoox.at/course/processing1`

Sobral R.S. (2021). Flipped classrooms for introductory computer programming courses. *International Journal of Information and Education Technology*, 11(4), 178–183. `http://www.ijiet.org/vol11/1508-AE002.pdf`

Webb, M. *et al.* (2017). computer science in the school curriculum: Issues and challenges. In: Tatnall, A., Webb, M. (eds) *Tomorrow's Learning: Involving Everyone. Learning with and about Technologies and Computing. WCCE 2017. IFIP Advances in Information and Communication Technology, vol 515.* Springer, Cham. `https://doi.org/10.1007/978-3-319-74310-3_43`

Wetzinger, E., Standl, B., Futschek, G. (2018). Developing a MOOC on introductory programming as additional preparation course for CS Freshmen. In: *EdMedia+ Innovate Learning*. Association for the Advancement of Computing in Education (AACE), pp. 1663–1672.

M. Landman – Researcher at TU Wien and member of the Informatics eduLAB group in the research unit of Information & Software Engineering since 2021. She is also a teacher and has experience in teaching computer science from 5th to 12th grade. She organizes the computer science faculty's school outreach activities, where she develops, organizes and conducts weekly workshops for school classes.



G. Futschek – is Professor at Institute of Information Systems Engineering and head of TU Wien Informatics eduLAB. His research area in informatics didactics is founded on his computer science background. His mission is to convey computational thinking and concepts of informatics in a way that makes fun and motivates to learn more. He does this in his lectures and with his group by outreaching to schools.

S. Unkovic – is a student assistant at TU Wien Informatics eduLAB since 2020. Currently she is completing her studies in mathematics and computer science. Her main work focuses on the creation and further development of creative materials for students and teachers. These materials are supposed to support them building a better understanding of computer science concepts.

F. Voboril – studies Software and Information Engineering at TU Wien, where she also works as student assistant. During her school years she attended courses in Robotics and participated in some competitions at this field. To share her knowledge, she teaches children in Robotics and Informatics. Since 2019 she is voluntary member in the Austrian team for the Bebras International Challenge on Informatics and Computational Thinking.